TUGAS AKHIR – IF184802

PENAMBAHAN ANOTASI OTOMATIS PADA DESKRIPSI MAKANAN UNTUK STUDI KASUS PENCARIAN KULINER

FADILLA SUKMA ALFIANI
NRP 05111640000024

Dosen Pembimbing 1
Nurul Fajrin Ariyani, S.Kom., M.Sc.

Dosen Pembimbing 2
Abdul Munif, S.Kom., M.Sc.

DEPARTEMEN TEKNIK INFORMATIKA
Fakultas Teknologi Elektro dan Informatika Cerdas
Institut Teknologi Sepuluh Nopember
Surabaya 2020

TUGAS AKHIR – IF184802

PENAMBAHAN ANOTASI OTOMATIS PADA DESKRIPSI MAKANAN UNTUK STUDI KASUS PENCARIAN KULINER

FADILLA SUKMA ALFIANI
NRP 05111640000024

Dosen Pembimbing 1
Nurul Fajrin Ariyani, S.Kom., M.Sc.

Dosen Pembimbing 2
Abdul Munif, S.Kom., M.Sc.

DEPARTEMEN TEKNIK INFORMATIKA
Fakultas Teknologi Elektro dan Informatika Cerdas
Institut Teknologi Sepuluh Nopember
Surabaya 2020

[Halaman ini sengaja dikosongkan]

UNDERGRADUATE THESIS – IF184802

AUTOMATED ANNOTATION FOR FOOD DESCRIPTION ON CASE STUDY CULINARY SEARCHING

FADILLA SUKMA ALFIANI
NRP 05111640000024

Supervisors 1
Nurul Fajrin Ariyani, S.Kom., M.Sc.

Supervisors 2
Abdul Munif, S.Kom., M.Sc.

Department of Informatics
Faculty of Intelligent Electrical and Informatics Technology
Institut Teknologi Sepuluh Nopember
Surabaya 2020

[Halaman ini sengaja dikosongkan]

LEMBAR PENGESAHAN

PENAMBAHAN ANOTASI OTOMATIS PADA DESKRIPSI MAKANAN UNTUK STUDI KASUS PENCARIAN KULINER

TUGAS AKHIR

Diajukan Guna Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Rumpun Mata Kuliah Manajemen Informasi
Program Studi S-1 Departemen Teknik Informatika
Fakultas Teknologi Elektro dan Informatika Cerdas
Institut Teknologi Sepuluh Nopember

Oleh:
FADILLA SUKMA ALFIANI
NRP 05111640000024

Disetujui oleh Dosen Pembimbing Tugas Akhir:

1. Nurul Fajrin Ariyani, S.Kom., M.Sc. .....
   NIP. 198607222015042003 (Pembimbing 1)

2. Abdul Munif., S.Kom., M.Sc. .....
   NIP. 198608232015041004 (Pembimbing 2)

KEMENTERIAN PENDIDIKAN DAN KEBUDAYAAN INSTITUT TEKNOLOGI SEPULUH NOPEMBER FAKULTAS TEKNOLOGI ELEKTRO DAN INFORMATIKA CERDAS DEPARTEMEN TEKNIK INFORMATIKA

SURABAYA
JANUARI 2020

[Halaman ini sengaja dikosongkan]

PENAMBAHAN ANOTASI OTOMATIS PADA DESKRIPSI MAKANAN UNTUK STUDI KASUS PENCARIAN KULINER

Nama Mahasiswa	: Fadilla Sukma Alfiani
NRP	: 05111640000024
Departemen	: Teknik Informatika ITS
Dosen Pembimbing 1	: Nurul Fajrin Ariyani, S.Kom., M.Sc.
Dosen Pembimbing 2	: Abdul Munif, S.Kom., M.Sc.

ABSTRAK

Di era modern ini, minat masyarakat terhadap kuliner terus meningkat. Untuk memenuhi keinginan konsumsi makanan yang sesuai selera, sebagian besar masyarakat menyatakan bahwa mereka mencari informasi tentang makanan terutama pada saat jam makan. Namun, sejauh ini search engine yang ada masih berdasarkan teks untuk mencari nama makanan saja. Padahal orang akan lebih mementingkan rasa makanan, bukan nama makanannya. Oleh karena itu, maka pada tugas akhir ini akan dibuat suatu aplikasi yang dapat membantu orang dalam melakukan pencarian kuliner yang diinginkan. Tidak hanya berdasarkan nama makanan saja, tetapi juga berdasarkan bahan, rasa, dan cara memasak makanan.

Dalam pembuatan aplikasi diperlukan deteksi entitas-entitas penting pada deskripsi makanan. Masalah ini dapat diatasi dengan menggunakan Named Entity Recognition (NER), Part-of-Speech (POS) Tagging, dan Rule-based Matching. Namun, dikarenakan model NER dan POS Tagging bahasa Indonesia milik SpaCy belum tersedia, maka pada tugas akhir ini juga dibuat model NER dan POS Tagging bahasa Indonesia baru dengan Prodigy sebagai alat bantu anotasinya. Selanjutnya hasil anotasi yang dilakukan dengan menggunakan model NER akan disimpan menjadi pasangan triplets pada triple store Apache Jena Fuseki.

Pada tugas akhir ini, ekstraksi setiap anotasi pada deskripsi makanan menjadi sebuah metadata dilakukan dengan mendeteksi

named entity berupa nama, bahan, rasa, dan cara memasak makanan menggunakan model NER. Hasil anotasi NER yang merupakan metadata makanan disimpan dalam bentuk pasangan triplets pada triple store Apache Jena Fuseki yang selanjutnya dapat digunakan untuk menjawab query tentang nama, bahan, rasa, dan cara memasak makanan.

Kata kunci: Anotasi, Klasifikasi, Kuliner

AUTOMATED ANNOTATION FOR FOOD DESCRIPTION ON CASE STUDY CULINARY SEARCHING

Student Name	: Fadilla Sukma Alfiani
NRP	: 05111640000024
Department	: Informatics ITS
Supervisor 1	: Nurul Fajrin Ariyani, S.Kom., M.Sc.
Supervisor 2	: Abdul Munif, S.Kom., M.Sc.

ABSTRACT

In this modern era, people's interest in culinary continues to increase. To meet the desires of food consumption according to taste, most people stated that they were looking for information about food, especially during meal times. However, so far the existing search engines are still based on text to search for food names only. Though people will be more concerned with the taste of food, not the name of the food. Therefore, this final project will create an application that can help people in their desired culinary search. Not only based on the name of the food, but also based on ingredients, taste, and how to cook food.

In making the application required the detection of important entities in the food description. This problem can be overcome by using Named Entity Recognition (NER), Part-of-Speech (POS) Tagging, and Rule-based Matching. However, because SpaCy's Indonesian NER and POS Tagging models are not yet available, this final project also created a new Indonesian NER and POS Tagging model with Prodigy as an annotation tool. Furthermore, the results of annotations performed using the NER model will be stored as a pair of triplets in the Apache Jena Fuseki triple store.

In this final project, the extraction of each annotation in the food description into metadata is done by detecting named entities in the form of names, ingredients, flavors, and how to cook food using the NER model. The NER annotation results which are food metadata are stored in the form of triplets in the Apache Jena

Fuseki triple store which can then be used to answer queries about names, ingredients, flavors, and how to cook food.

Keywords: Annotation, Classification, Culinary

KATA PENGANTAR

Puji syukur saya ucapkan kepada Allah Yang Maha Kuasa. Karena atas karunia serta rahmat-Nya saya dapat menyelesaikan tugas akhir yang berjudul:

PENAMBAHAN ANOTASI OTOMATIS PADA DESKRIPSI MAKANAN UNTUK STUDI KASUS PENCARIAN KULINER

Selesainya tugas akhir ini tidak lepas dari bantuan dan dukungan beberapa pihak. Sehingga pada kesempatan ini penulis mengucapkan syukur dan terima kasih kepada:

1. Allah SWT Yang Maha Kuasa, karena limpahan rahmat dan karunia-Nya penulis dapat menjalankan perkuliahan di Departemen Informatika Institut Teknologi Sepuluh Nopember dan dapat menyelesaikan tugas akhir guna memenuhi syarat kelulusan sebagai Sarjana.
2. Keluarga tercinta, Bapak dan Ibu yang telah memberikan dukungan moral dan material serta doa yang tak terhingga untuk penulis.
3. Ibu Nurul Fajrin Ariyani, S.Kom., M.Sc. selaku dosen wali dan pembimbing I yang telah memberi dukungan, semangat, dan motivasi selama masa perkuliahan penulis, serta mengorbankan waktu dan tenaga untuk membimbing penulis dalam menyelesaikan tugas akhir ini.
4. Bapak Abdul Munif, S.Kom., M.Sc. selaku pembimbing II yang telah memberi masukan dan saran serta mengorbankan waktu dan tenaga untuk membimbing penulis dalam menyelesaikan tugas akhir ini.
5. Segenap dosen dan karyawan Departemen Informatika ITS yang telah memberikan ilmu dan pengalaman kepada penulis selama menjalani masa kuliah di Informatika ITS.
6. Desy Nilasari, Denise Sonia Rahmadina, dan Akbar Noto Ponco Bimantoro sebagai teman seperjuangan tugas akhir

yang selalu memberi semangat dan bantuan dalam pengerjaan tugas akhir ini.

7. Ismail Syarif dan Nuzha Musyafira yang membantu penulis dalam pengerjaan tugas akhir ini.
8. Dewi Sekarini, Ivanda Zevi Amalia, dan Akbar Noto Ponco Bimantoro sebagai teman seperjuangan *fast track* yang saling memberi dukungan dan semangat dalam menghadapi perkuliahan.
9. Denise, Nila, Dewi, dan Diana yang berjuang bersama dan menemani penulis dalam pengerjaan tugas akhir ini.
10. Teman-teman Informatika ITS angkatan 2016 yang tidak bisa disebutkan namanya satu persatu.
11. Serta semua pihak yang yang telah turut membantu penulis dalam menyelesaikan tugas akhir ini.

Penulis memohon maaf jika terdapat kesalahan maupun kekurangan pada tugas akhir ini. Oleh karena itu, dengan segala kerendahan hati penulis mengharapkan kritik dan saran yang membangun dapat disampaikan sebagai bahan perbaikan untuk kedepannya. Semoga laporan tugas akhir ini dapat berguna bagi pembaca.

Surabaya, Januari 2020

Fadilla Sukma Alfiani

DAFTAR ISI

[Halaman ini sengaja dikosongkan]

DAFTAR GAMBAR

DAFTAR TABEL

[Halaman ini sengaja dikosongkan]

DAFTAR KODE SUMBER

[Halaman ini sengaja dikosongkan]

BAB I
PENDAHULUAN

1.1 Latar Belakang

Di era modern ini, minat masyarakat terhadap kuliner terus meningkat. Untuk memenuhi keinginan konsumsi makanan yang sesuai selera, sebagian besar masyarakat menyatakan bahwa mereka mencari informasi tentang makanan terutama pada saat jam makan [1]. Namun, sejauh ini *search engine* yang ada masih berdasarkan teks untuk mencari nama makanan saja. Padahal orang akan lebih mementingkan rasa makanan, bukan nama makanannya.

Oleh karena hal di atas, maka pada tugas akhir ini akan dibuat suatu aplikasi yang dapat membantu orang dalam melakukan pencarian kuliner yang diinginkan. Pencarian tidak hanya berdasarkan nama makanan saja, tetapi juga berdasarkan bahan, rasa, dan cara memasak makanan. Pembuatan aplikasi dilakukan dengan menambahkan anotasi pada deskripsi makanan. Penambahan anotasi pada deskripsi makanan akan memudahkan orang dalam mencari kuliner yang diinginkan karena proses pencarian kuliner berdasarkan nama, bahan, rasa, dan cara memasak makanan menjadi jauh lebih cepat. Namun, banyak kendala yang dihadapi jika penambahan anotasi pada deskripsi makanan dilakukan secara manual. Anotasi yang dilakukan secara manual sangat tidak praktis, sulit diimplementasikan, dan memakan banyak waktu. Oleh karena itu, diperlukan suatu cara untuk melakukan penambahan anotasi secara otomatis pada deskripsi makanan seperti yang akan dibahas pada tugas akhir ini.

1.2 Rumusan Masalah

Rumusan masalah yang diangkat pada tugas akhir ini adalah sebagai berikut:

1. Bagaimana cara mengekstraksi setiap anotasi pada deskripsi makanan menjadi sebuah metadata?

2. Bagaimana cara memodelkan metadata makanan yang dapat menjawab *query* tentang bahan, rasa, dan cara memasak makanan?

1.3 Batasan Masalah

Batasan masalah pada tugas akhir ini antara lain:

1. Deskripsi makanan yang digunakan merupakan deskripsi yang memiliki nama, bahan, rasa, dan cara memasak makanan.
2. Ruang lingkup untuk data masukan terbatas pada nama, bahan, rasa, dan cara memasak makanan.
3. Dataset yang dipilih menggunakan bahasa Indonesia.
4. Aplikasi hanya berfokus pada klasifikasi dan penambahan anotasi (berupa label), sehingga tidak dapat dilakukan *reasoning*.

1.4 Tujuan

Tujuan dari pembuatan tugas akhir ini adalah sebagai berikut:

1. Dapat mengekstraksi setiap anotasi pada deskripsi makanan menjadi sebuah metadata
2. Dapat memodelkan metadata makanan yang dapat menjawab *query* tentang bahan, rasa, dan cara memasak makanan

1.5 Manfaat

Manfaat dari pembuatan tugas akhir ini adalah untuk menghasilkan suatu aplikasi yang dapat membantu orang dalam melakukan pencarian kuliner yang diinginkan dengan melakukan penambahan anotasi otomatis pada deskripsi makanan yang tersedia sehingga pengguna akan lebih mudah mengenali makanan yang diinginkan. Data deskripsi makanan tersebut diklasifikasi dan dimaknai sebagai nama, bahan, rasa, dan cara memasak makanan.

1.6 Metodologi

Langkah-langkah yang harus ditempuh dalam pengerjaan tugas akhir ini adalah sebagai berikut:

1. Penyusunan Proposal Tugas Akhir

Proposal tugas akhir berisi tentang penjelasan mengenai pendahuluan dari tugas akhir yang dibuat. Pendahuluan ini terdiri atas deskripsi latar belakang, rumusan masalah, batasan masalah, tujuan, dan manfaat dari tugas akhir. Selain itu, dijelaskan pula tinjauan pustaka yang digunakan sebagai referensi dalam pengerjaan tugas akhir. Pada subbab metodologi dijelaskan mengenai urutan pengerjaan tugas akhir dimulai dari penyusunan proposal sampai penyusunan buku tugas akhir.

2. Studi Literatur

Pada studi literatur akan dipelajari beberapa referensi yang relevan terhadap tugas akhir yang dikerjakan. Literasi yang didapatkan berasal dari jurnal yang kredibel, dokumentasi website resmi, dan artikel yang dapat dipercaya. Hal-hal yang dipelajari yaitu tentang kuliner, *Natural Language Processing*, *text processing*, *Named Entity Recognition* (NER), *POS Tagging*, dan sebagainya.

3. Analisis dan desain sistem

Analisis yang dilakukan pada tugas akhir ini adalah dengan memahami dataset yang telah diambil. Kemudian dilakukan klasifikasi dengan *Machine Learning*. Desain sistem nantinya *user* dapat melakukan *query* atau pencarian makanan berdasarkan nama, bahan, rasa, dan cara memasak makanan. Selain itu pengguna juga dapat melakukan *input*/*supply* data makanan berupa nama dan deskripsi makanan.

4. Implementasi

Kakas bantu yang digunakan dalam penulisan program pada tugas akhir ini berupa bahasa pemrograman Python dan IDE Jupyter Notebook. Pustaka yang digunakan merupakan pustaka bawaan Python, antara lain Flask. spaCy, dan pustaka lain yang mendukung *text processing* mulai *pre-processing* hingga pengujian dan evaluasi.

Sedangkan rincian perangkat keras dan sistem operasi yang digunakan selama tugas akhir adalah:

- Processor Intel® Core™ i7-4720HQ CPU @ 2.60GHz
- Installed RAM 12.0 GB (11.9 GB usable)
- System Type 64-bit operating system, x64-based processor
- Windows Edition Windows 8.1 Pro

5. Uji Coba dan Evaluasi

Uji coba pada tugas akhir ini dilakukan untuk mengetahui apakah penggunaan metode sudah tepat demi tercapainya tujuan dari tugas akhir ini. Sebelum pengujian dilakukan, terlebih dahulu harus dibuat skenario uji agar pengujian tepat sasaran dan sesuai dengan tujuan yang diinginkan.

Evaluasi pada tugas akhir ini dilakukan untuk mendapatkan nilai kuantitatif dari skenario uji yang dilakukan. Metrik yang digunakan adalah nilai *accuracy*, *precision*, *recall*, dan *F-Measure*. Hasil metrik akan dinterpretasikan untuk menjawab apakah tujuan dari tugas akhir ini tercapai.

6. Penyusunan Buku Tugas Akhir

Pada tahap ini dilakukan penyusunan laporan yang menjelaskan dasar teori dan metode yang digunakan pada tugas akhir ini serta hasil dari implementasi aplikasi perangkat lunak yang telah dibuat. Sistematika penulisan buku tugas akhir secara garis besar antara lain:

1. Pendahuluan
    a. Latar Belakang
    b. Rumusan Masalah
    c. Batasan Tugas Akhir
    d. Tujuan
    e. Manfaat
    f. Metodologi
    g. Sistematika Penulisan
2. Tinjauan Pustaka
3. Analisis dan Perancangan
4. Implementasi

5. Uji Coba dan Evaluasi
6. Kesimpulan dan Saran
7. Daftar Pustaka

1.7 Sistematika Penulisan

Sistematika Penulisan Buku tugas akhir ini bertujuan untuk mendapatkan gambaran dari pengerjaan tugas akhir. Selain itu, diharapkan dapat berguna untuk pembaca yang tertarik untuk melakukan pengembangan lebih lanjut. Secara garis besar, buku tugas akhir terdiri atas beberapa bagian seperti berikut ini:

BAB I. **Pendahuluan**
Bab ini berisi latar belakang masalah, tujuan dan manfaat pembuatan tugas akhir, permasalahan, batasan masalah, metodologi yang digunakan, dan sistematika penyusunan tugas akhir.

BAB II. **Tinjauan Pustaka**
Bab ini menjelaskan beberapa teori yang dijadikan penunjang dan berhubungan dengan pokok pembahasan yang mendasari pembuatan tugas akhir.

BAB III. **Analisis dan Perancangan**
Bab ini membahas mengenai perancangan sistem yang akan dibangun. Perancangan sistem meliputi perancangan data dan alur proses dari sistem itu sendiri.

BAB IV. **Implementasi**
Bab ini berisi implementasi dari perancangan sistem yang telah ditentukan sebelumnya.

BAB V. **Uji Coba dan Evaluasi**
Bab ini membahas pengujian dari metode yang ditawarkan pada tugas akhir untuk mengetahui kesesuaian metode dengan data yang ada.

BAB VI. Kesimpulan dan Saran

Kesimpulan bab ini berisi kesimpulan dari hasil pengujian yang telah dilakukan. Bab ini juga membahas saran-saran untuk pengembangan sistem lebih lanjut.

Daftar Pustaka

Merupakan daftar referensi yang digunakan untuk mengembangkan tugas akhir.

Lampiran

Merupakan bab tambahan yang berisi data atau daftar istilah yang penting pada tugas akhir ini.

BAB II
TINJAUAN PUSTAKA

Pada bab ini akan dibahas mengenai teori-teori yang menjadi dasar tugas akhir ini.

2.1 Penelitian Terkait

Penelitian terkait yang sudah dilakukan sebelumnya berfokus pada pembangunan ontologi baru untuk sebuah *platform* yang menghubungkan orang-orang yang memiliki makanan berlebih dengan mereka yang membutuhkannya. Ontologi yang dibangun berguna untuk memberikan informasi kepada pengguna mengenai nama makanan, cara penyajian makanan, dan kandungan bahan pada suatu makanan tersebut [2].

Kaitan penelitian tersebut dengan tugas akhir ini adalah pada implementasi penyimpanan data yang menggunakan ontologi. Penelitian tersebut berfokus pada pembangunan ontologi dari awal karena harus menyesuaikan dengan studi kasus yang diangkat. Ontologi dibuat secara manual dengan memasukkan satu per satu kelas, *instance* (individual), dan *property* (atribut). Sedangkan pada tugas akhir ini, ontologi awal yang dibuat hanya berisi inisiasi kelas dan *property* baru. Untuk pengisian *instance* dan *property* dilakukan secara otomatis melalui aplikasi yang dibangun pada tugas akhir ini.

Penelitian terkait selanjutnya berfokus pada pembangunan *chatbot* untuk salah satu *start-up* yang bergerak untuk berbagi makanan yang berlebih kepada orang yang membutuhkan. *Chatbot* tersebut dibangun untuk membantu konsumen dalam melakukan transaksi berbagi makanan dan mencari makanan sesuai yang diinginkan. Secara umum, *chatbot* ini dibangun pada *platform* web menggunakan *Natural Language Processing* (NLP) untuk mengubah bahasa alami menjadi *query*, ontologi untuk menyimpan data, dan menambahkan *speech recognition* sebagai salah satu fitur untuk konsumen agar lebih interaktif dengan *chatbot* [3].

Kaitan penelitian tersebut dengan tugas akhir ini hampir sama dengan penelitian yang pertama yaitu tentang implementasi penyimpanan data menggunakan ontologi. Pada penelitian kedua ontologi yang dibangun menyimpan kelas makanan yang nantinya digunakan dalam proses *query* atau pencarian makanan. Namun, pengguna *chatbot* terbatas hanya dapat melakukan pencarian nama makanan saja. Sedangkan pada tugas akhir ini ontologi yang dibuat tidak hanya memiliki kelas makanan saja. Ontologi juga digunakan untuk menyimpan entitas bahan, rasa, dan cara memasak makanan. Sehingga nantinya dalam proses pencarian makanan pengguna tidak hanya dapat melakukan pencarian berdasarkan nama makanan saja tetapi bisa juga berdasarkan bahan, rasa, cara memasak, atau bahkan gabungan dua atau lebih dari keempatnya.

2.2 Kuliner

Kuliner adalah suatu bagian hidup yang erat kaitannya dengan konsumsi makanan sehari-hari karena setiap orang memerlukan makanan yang sangat dibutuhkan sehari-hari. Mulai dari makanan yang sederhana hingga makanan yang berkelas tinggi dan mewah. Sementara istilah kuliner merupakan unsur serapan dari bahasa Inggris yaitu *culinary* yang berarti urusan masak memasak.

Gambar 2.1 Contoh laman Wikipedia

Dengan terus berkembangnya teknologi dan kemudahan berbagi informasi secara *online*, saat ini banyak orang yang mencari informasi seputar kuliner pada blog atau *website* kuliner dengan tujuan mendapatkan informasi kuliner yang diinginkan, rekomendasi wisata kuliner, serta berbagai jenis resep masakan.

Gambar 2.2 Contoh laman blog kuliner

Pada tugas akhir ini data kuliner yang digunakan merupakan nama makanan dan kalimat-kalimat deskripsi makanan yang diambil dari Wikipedia dan berbagai blog yang memuat aneka makanan khas di Indonesia, seperti salah satu contohnya adalah Blog Tokopedia. Contoh laman Wikipedia yang memuat informasi tentang makanan ditunjukkan oleh Gambar 2.1 dan contoh laman

blog kuliner ditunjukkan oleh Gambar 2.2. Dari deskripsi makanan yang diambil sebagian besar memuat bahan, rasa, dan cara memasak makanan. Ketiga entitas tersebut paling sering disebutkan dalam deskripsi makanan. Sehingga pada tugas akhir ini, ruang lingkup aplikasi pencarian kuliner yang dibuat berfokus pada nama, bahan, rasa, dan cara memasak makanan.

2.3 Korpus

Korpus merupakan sekumpulan materi tertulis berupa teks yang menjadi dasar analisis linguistik [4]. Berdasarkan pengertian tersebut, dapat dikatakan bahwa korpus terutama muncul dalam area NLP (*Natural Language Processing*) atau domain aplikasi yang berkaitan dengan teks atau dokumen. Pada tugas akhir ini korpus yang digunakan adalah korpus data makanan.

2.4 Natural Language Processing

NLP (*Natural Language Processing*) merupakan salah satu cabang ilmu kecerdasan buatan (*artificial intelligence*) yang berfokus pada pengolahan bahasa natural. Bahasa natural adalah bahasa yang secara umum digunakan oleh manusia dalam berkomunikasi satu sama lain. Bahasa yang diterima oleh komputer butuh untuk diproses dan dipahami terlebih dahulu supaya maksud dari user bisa dipahami dengan baik oleh komputer.

Beberapa bidang penerapan NLP antara lain penjawab pertanyaan (*question answering*), ekstraksi informasi (*information extraction*), analisis sentimen (*sentiment analysis*), penerjemahan mesin (*machine translation*), pemerolehan informasi (*information retrival*), perangkuman otomatis (*automatic summarization*), dan pengenalan wicara (*speech recognition*). Bidang penerapan NLP yang digunakan pada tugas akhir ini adalah *information extraction* dimana program dapat mengekstrak data menjadi informasi yang dapat dimanfaatkan.

2.5 *Text Processing*

Text Processing merupakan tahap awal dimana sistem melakukan seleksi data yang akan diproses, dengan tujuan mempersiapkan data agar menjadi lebih terstruktur. Berikut merupakan beberapa tahapan dalam *text processing*. Tahapan tersebut dilakukan secara berurutan. *Output* pada suatu tahapan akan menjadi *input* pada tahapan selanjutnya.

2.5.1 *Case Folding*

Case folding adalah mengubah semua huruf dalam dokumen menjadi huruf kecil (*lower case*). Hanya huruf ‘a’ sampai dengan ‘z’ yang diterima. Karakter selain huruf dihilangkan dan dianggap delimiter. Contoh *case folding* dapat dilihat pada Tabel 2.1.

Tabel 2.1 Contoh *case folding*

Teks Awal	**Hasil *Case Folding***
Gudeg adalah makanan khas Jogja yang terbuat dari nangka muda yang dimasak dengan santan. Makanan khas Jogja ini memiliki warna coklat yang biasanya dihasilkan oleh daun jati yang dimasak bersamaan. Gudeg biasanya dimakan dengan nasi dan disajikan dengan kuah santan kental (areh), ayam kampung, telur, tahu dan sambal goreng krecek.	gudeg adalah makanan khas jogja yang terbuat dari nangka muda yang dimasak dengan santan. makanan khas jogja ini memiliki warna coklat yang biasanya dihasilkan oleh daun jati yang dimasak bersamaan. gudeg biasanya dimakan dengan nasi dan disajikan dengan kuah santan kental (areh), ayam kampung, telur, tahu dan sambal goreng krecek.

2.5.2 Penghapusan Tanda Baca

Penghapusan tanda baca (*punctuation*) digunakan untuk mempermudah analisis, karena dalam melakukan *text processing*, tanda baca dianggap sebagai *noise*. Contoh penghapusan tanda baca dapat dilihat pada Tabel 2.2.

Tabel 2.2 Contoh penghapusan tanda baca

Teks Awal	Hasil Penghapusan Tanda Baca
gudeg adalah makanan khas jogja yang terbuat dari nangka muda yang dimasak dengan santan. makanan khas jogja ini memiliki warna coklat yang biasanya dihasilkan oleh daun jati yang dimasak bersamaan. gudeg biasanya dimakan dengan nasi dan disajikan dengan kuah santan kental (areh), ayam kampung, telur, tahu dan sambal goreng krecek.	gudeg adalah makanan khas jogja yang terbuat dari nangka muda yang dimasak dengan santan makanan khas jogja ini memiliki warna coklat yang biasanya dihasilkan oleh daun jati yang dimasak bersamaan gudeg biasanya dimakan dengan nasi dan disajikan dengan kuah santan kental areh ayam kampung telur tahu dan sambal goreng krecek

2.5.3 Tokenisasi

Tokenisasi merupakan tahap pemotongan *string input* berdasarkan tiap kata yang menyusunnya. Contoh tokenisasi dapat dilihat pada Tabel 2.3.

Tabel 2.3 Contoh tokenisasi

Teks Awal	Hasil Tokenisasi
gudeg adalah makanan khas jogja yang terbuat dari nangka	'gudeg', 'adalah', 'makanan', 'khas', 'jogja', 'yang', 'terbuat',

Teks Awal	Hasil Tokenisasi
muda yang dimasak dengan santan makanan khas jogja ini memiliki warna coklat yang biasanya dihasilkan oleh daun jati yang dimasak bersamaan gudeg biasanya dimakan dengan nasi dan disajikan dengan kuah santan kental areh ayam kampung telur tahu dan sambal goreng krecek	'dari', 'nangka', 'muda', 'yang', 'dimasak', 'dengan', 'santan', 'makanan', 'khas', 'jogja', 'ini', 'memiliki', 'warna', 'coklat', 'yang', 'biasanya', 'dihasilkan', 'oleh', 'daun', 'jati', 'yang', 'dimasak', 'bersamaan', 'gudeg', 'biasanya', 'dimakan', 'dengan', 'nasi', 'dan', 'disajikan', 'dengan', 'kuah', 'santan', 'kental', 'areh', ' ', 'ayam', 'kampung', 'telur', 'tahu', 'dan', 'sambal', 'goreng', 'krecek'

2.5.4 *Stopwords Removal*

Stopwords merupakan kata umum yang biasanya muncul dalam jumlah besar dan dianggap tidak memiliki makna. Tujuan melakukan *stopwords removal* adalah untuk menghilangkan kata yang tidak memiliki makna dan mengurangi jumlah kata yang akan diproses. Contoh *stopwords removal* dapat dilihat pada Tabel 2.4.

Tabel 2.4 Contoh *stopword removal*

Teks Awal	Hasil *Stopword Removal*
'gudeg', 'adalah', 'makanan', 'khas', 'jogja', 'yang', 'terbuat', 'dari', 'nangka', 'muda', 'yang', 'dimasak', 'dengan', 'santan', 'makanan', 'khas', 'jogja', 'ini', 'memiliki', 'warna', 'coklat', 'yang', 'biasanya', 'dihasilkan', 'oleh', 'daun', 'jati', 'yang',	gudeg makanan khas jogja terbuat nangka muda dimasak santan makanan khas jogja memiliki warna coklat dihasilkan daun jati dimasak bersamaan gudeg dimakan nasi disajikan kuah santan

Teks Awal	Hasil *Stopword Removal*
'dimasak', 'bersamaan', 'gudeg', 'biasanya', 'dimakan', 'dengan', 'nasi', 'dan', 'disajikan', 'dengan', 'kuah', 'santan', 'kental', 'areh', ' ', 'ayam', 'kampung', 'telur', 'tahu', 'dan', 'sambal', 'goreng', 'krecek'	kental areh ayam kampung telur sambal goreng krecek

2.6 *Named Entity Recognition* (NER)

Named Entity Recognition (NER) merupakan bagian dari ekstraksi informasi yang bertugas untuk mengklasifikasikan teks dari sebuah dokumen atau korpus yang dikategorikan seperti nama orang, lokasi, organisasi, tanggal, waktu, dan sebagainya. NER diimplementasikan dalam banyak bidang, antara lain dalam *machine translation*, *question-answering machine system*, *indexing* pada *information retrieval*, klasifikasi, dan juga dalam *automatic summarization*. Tujuan yang diharapkan dari proses dalam NER adalah untuk melakukan ekstraksi dan klasifikasi nama ke dalam beberapa kategori dengan mengacu kepada makna yang tepat [5]. Pada tugas akhir ini NER digunakan untuk mengenali entitas nama, bahan, rasa, dan cara memasak makanan dari deskripsi makanan. Contoh NER dapat dilihat pada Tabel 2.5.

Tabel 2.5 Contoh NER

Teks Awal	Hasil NER
Gudeg adalah makanan khas Jogja yang terbuat dari nangka muda yang dimasak dengan santan. Makanan khas Jogja ini memiliki warna coklat yang biasanya	gudeg NAMA nangka muda BAHAN santan BAHAN daun jati BAHAN santan kental BAHAN areh BAHAN

Teks Awal	Hasil NER
dihasilkan oleh daun jati yang dimasak bersamaan. Gudeg biasanya dimakan dengan nasi dan disajikan dengan kuah santan kental (areh), ayam kampung, telur, tahu dan sambal goreng krecek.	ayam kampung BAHAN telur BAHAN tahu BAHAN sambal goreng krecek BAHAN

2.7 POS *Tagging*

Part of Speech (POS) *Tagging* adalah proses memberi label pada setiap kata dalam kalimat dengan POS atau *tag* yang sesuai dengan kelas kata seperti kata benda, kata kerja, kata keterangan, kata sifat, dan lainnya [6]. Pada tugas akhir ini POS *Tagging* digunakan untuk mencari kelas tiap kata dari deskripsi makanan untuk kemudian digunakan dalam proses *Rule-based Matching*. Contoh POS *Tagging* dapat dilihat pada Tabel 2.6.

Tabel 2.6 Contoh POS *Tagging*

Teks Awal	Hasil POS *Tagging*
Gudeg adalah makanan khas Jogja yang terbuat dari nangka muda yang dimasak dengan santan.	gudeg NOUN adalah VERB makanan NOUN khas NOUN yogyakarta PROPN yang SCONJ terbuat VERB dari ADP nangka NOUN muda ADJ yang SCONJ dimasak VERB

Teks Awal	Hasil POS *Tagging*
	dengan ADP santan NOUN . PUNCT

2.8 *Rule-based Matching*

Rule-based Matching membantu pengguna untuk mencocokkan token, frasa, entitas kata, dan kalimat berdasarkan beberapa pola yang ditentukan. Proses ini dilakukan bersama dengan fitur lain seperti *parts-of-speech* [7]. Pada tugas akhir ini *Rule-based Matching* digunakan untuk mencari entitas bahan dari deskripsi makanan berdasarkan *rule* yang telah dibuat.

2.9 SpaCy

SpaCy adalah pustaka *open-source* gratis dalam bahasa Python yang digunakan untuk *Natural Language Processing* (NLP). SpaCy dapat digunakan untuk membangun ekstraksi informasi atau praproses teks untuk *deep learning*. Beberapa fitur yang didukung spaCy antara lain tokenisasi, POS *Tagging*, *Named Entity Recognition*, dan *Rule-based Matching* [8]. Pada tugas akhir ini spaCy digunakan untuk melakukan anotasi NER, POS *Tagging*, dan *Rule-based Matching*.

2.10 Prodigy

Prodigy merupakan alat anotasi yang efisien sehingga penggunanya dapat melakukan anotasi sendiri. Pengguna dapat bekerja pada pengenalan entitas, dan Prodigy dapat membantu pengguna dalam melatih dan mengevaluasi model dengan lebih cepat. Fitur yang ada dalam Prodigy yaitu *Named Entity Recognition*, klasifikasi teks, dan visi komputer [9]. Pada tugas akhir ini Prodigy digunakan sebagai alat bantu untuk melakukan anotasi NER pada deskripsi makanan.

2.11 Flask

Flask merupakan salah satu kerangka kerja aplikasi web Python yang paling populer. Flask memberikan kebebasan kepada pengembang untuk memilih alat dan pustaka yang ingin digunakan. Serta ada banyak ekstensi yang disediakan sehingga penambahan fungsionalitas baru menjadi lebih mudah [10]. Pada tugas akhir ini Flask digunakan untuk membuat aplikasi web.

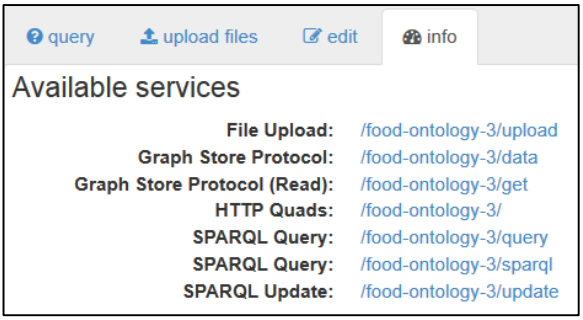
2.12 SPARQL

SPARQL merupakan protokol bahasa *query* RDF yang berfungsi untuk mengambil dan memanipulasi data dari sebuah *triple store*. RDF (*Resource Description Framework*) merupakan jenis *file* yang terdiri dari *statement* yang memiliki 3 variabel sebagai subjek, predikat, objek, yang dikenal dengan sebutan *triplets*. Pada tugas akhir ini, SPARQL *query* digunakan untuk mencari nama makanan yang tersimpan dalam *triple store* Apache Jena Fuseki dan untuk meng-*input* data makanan beserta anotasinya pada *triple store* Apache Jena Fuseki.

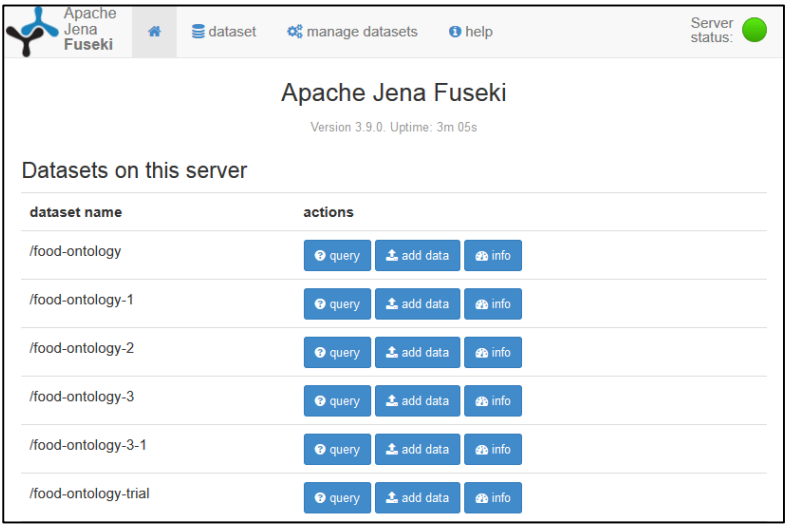
2.13 Apache Jena Fuseki

Apache Jena Fuseki adalah server SPARQL (bahasa *query* semantik). Apache Jena Fuseki memiliki antarmuka pengguna untuk pemantauan dan administrasi server. Selain itu, Apache Jena Fuseki juga menyediakan mesin protokol untuk sistem penyimpanan dan *query* RDF (*Resource Description Framework*) [11]. Pada tugas akhir ini Apache Jena Fuseki bertindak sebagai basis data *triple store* yang bisa diakses melalui *request* HTTP. Gambar 2.4 menunjukkan daftar *dataset* yang ada di dalam server Apache Jena Fuseki.

Apache Jena Fuseki menyediakan beberapa API seperti ditunjukkan pada Gambar 2.3. Beberapa fungsi API tersebut antara lain untuk *endpoint* pengunggahan file RDF (`/upload`), membaca data (`/get`), *query* SPARQL (`/query` atau `/sparql`), dan memperbarui data (`/update`).



Gambar 2.3 Daftar API Apache Jena Fuseki



Gambar 2.4 *Triple store* Apache Jena Fuseki

2.14 *Confusion Matrix*

Confusion matrix merupakan salah satu metode yang dapat digunakan untuk mengukur kinerja suatu metode klasifikasi. Pada dasarnya, *confusion matrix* mengandung informasi yang

membandingkan hasil klasifikasi yang dilakukan oleh sistem dengan hasil klasifikasi yang seharusnya.

Pada pengukuran kinerja menggunakan *confusion matrix*, terdapat 4 (empat) istilah sebagai representasi hasil proses klasifikasi. Keempat istilah tersebut adalah *True Positive* (TP), *True Negative* (TN), *False Positive* (FP), dan *False Negative* (FN) [12].

True Positive (TP) menyatakan jumlah kebenaran antara hasil klasifikasi dengan jumlah seluruh data. *True Negative* (TN) menyatakan jumlah hasil klasifikasi yang diindikasikan benar, tetapi sesungguhnya salah. *False Positive* (FP) menyatakan jumlah dari hasil klasifikasi yang diindikasikan salah, tetapi sesungguhnya benar. *False Negative* (FN) menyatakan jumlah dari kesamaan hasil klasifikasi dan yang sesungguhnya adalah salah.

		Actual Values	
		Positive (1)	Negative (0)
Predicted Values	Positive (1)	TP	FP
	Negative (0)	FN	TN

Gambar 2.5 *Confusion Matrix*

Metrik evaluasi *accuracy*, *precision*, *recall*, dan *F-Measure* secara lebih rinci dijelaskan sebagaimana berikut:

a. *Accuracy*

Accuracy adalah adalah nilai rasio data yang diklasifikasikan benar dari jumlah total data.

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN} \quad (2.15)$$

b. *Precision*

Precision adalah nilai total data positif yang diklasifikasikan dengan benar dibagi dengan hasil prediksi data positif.

$$Precision = \frac{TP}{TP + FP} \quad (2.16)$$

c. *Recall*

Recall adalah nilai total data positif yang diklasifikasikan dengan benar dibagi dengan jumlah data positif.

$$Recall = \frac{TP}{TP + FN} \quad (2.17)$$

d. *F-Measure*

F-Measure adalah nilai yang didapatkan dari *precision* dan *recall* menggunakan *Harmonic Mean*.

$$F_1 = 2 \times \frac{precision \times recall}{precision + recall} \quad (2.18)$$

Kemudian, hasil dari masing-masing perspektif pada setiap kelas dirata-rata untuk mendapatkan hasil akhir dari *accuracy*, *precision*, *recall*, dan *F-Measure*.

BAB III
ANALISIS DAN PERANCANGAN

Pada bab ini akan dijabarkan analisis dan perancangan sistem dari tugas akhir yang meliputi tahap-tahap penyelesaian tugas akhir dan algoritma-algoritma yang mendukungnya secara khusus.

3.1 Analisis Permasalahan

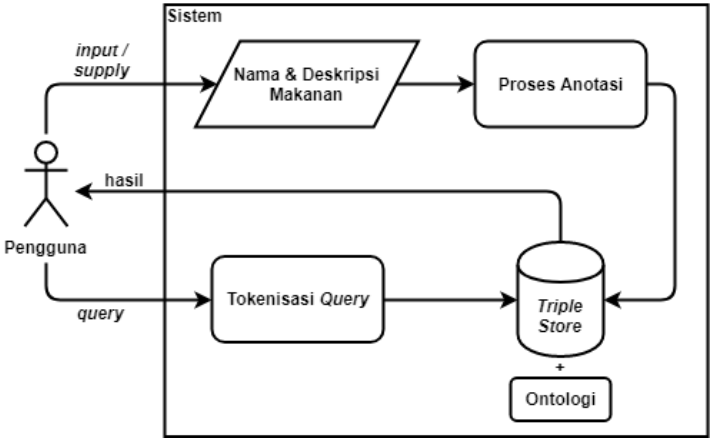
Pokok permasalahan yang akan dibahas dan akan dicarikan solusinya dalam tugas akhir ini adalah penambahan anotasi otomatis pada deskripsi makanan untuk studi kasus pencarian kuliner. Dalam kehidupan sehari-hari sebagian besar orang memenuhi keinginan konsumsi makanan mereka yang sesuai selera adalah dengan mencari informasi tentang makanan. Namun, sejauh ini *search engine* yang ada masih berdasarkan teks untuk mencari nama makanan saja. Padahal kemungkinan besar orang akan lebih mementingkan rasa atau bahan makanan, bukan nama makanannya.

Dengan melihat permasalahan di atas, maka pada tugas akhir ini akan dibuat suatu aplikasi yang dapat membantu orang dalam melakukan pencarian kuliner yang diinginkan. Pencarian tidak hanya berdasarkan nama makanan saja, tetapi juga berdasarkan bahan, rasa, dan cara memasak makanan. Pembuatan aplikasi dilakukan dengan menambahkan anotasi pada deskripsi makanan. Penambahan anotasi pada deskripsi makanan akan memudahkan orang dalam mencari kuliner yang diinginkan karena proses pencarian kuliner berdasarkan nama, bahan, rasa, dan cara memasak makanan menjadi jauh lebih cepat. Namun, banyak kendala yang dihadapi jika penambahan anotasi pada deskripsi makanan dilakukan secara manual. Anotasi yang dilakukan secara manual sangat tidak praktis, sulit diimplementasikan, dan memakan banyak waktu. Oleh karena itu, diperlukan suatu cara untuk melakukan penambahan anotasi secara otomatis pada deskripsi makanan seperti yang akan dibahas pada tugas akhir ini.

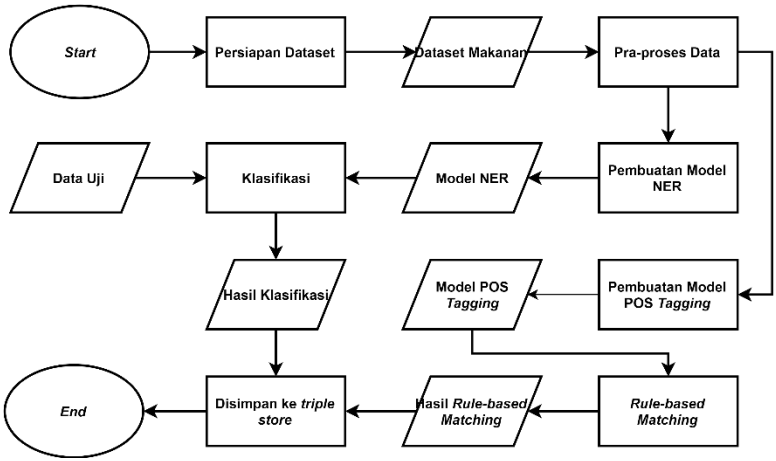
3.2 Deskripsi Umum Sistem

Pada tugas akhir ini akan dibangun suatu aplikasi yang dapat membantu orang dalam melakukan pencarian kuliner yang diinginkan dengan melakukan penambahan anotasi otomatis pada deskripsi makanan yang tersedia. Data deskripsi makanan tersebut diklasifikasi dan dimaknai sebagai nama, bahan, rasa, dan cara memasak makanan.

Arsitektur sistem yang dirancang pada tugas akhir ini ditunjukkan oleh Gambar 3.1. Pengguna dapat melakukan *input*/*supply* data serta *query* atau pencarian makanan. *Input* data harus berupa nama dan deskripsi makanan yang kemudian masuk ke proses anotasi otomatis dan disimpan ke dalam *triple store*. Sedangkan pencarian makanan atau *query* dilakukan dengan memasukkan kata kunci pencarian. Kemudian sistem akan memotong tiap kata kunci pencarian lalu mengambil data yang sesuai dari *triple store*.



Gambar 3.1 Arsitektur sistem



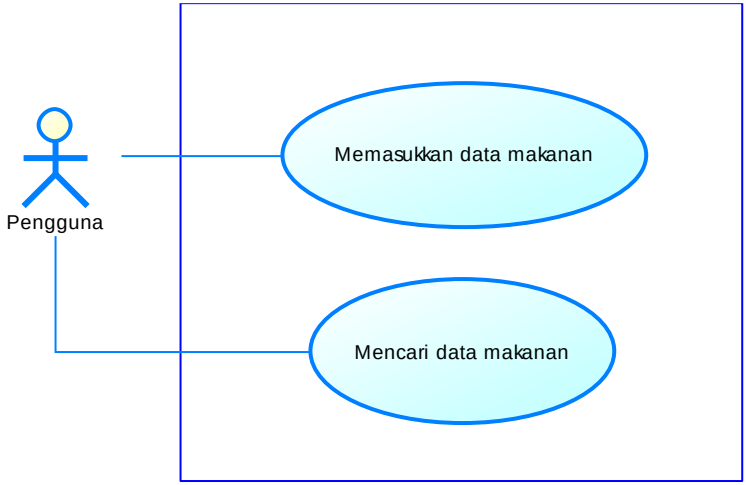
Gambar 3.2 Diagram alir seluruh tahapan

Pada diagram alir seluruh tahapan seperti ditunjukkan oleh Gambar 3.2 digambarkan bahwa tahap pertama dilakukan persiapan dataset dengan cara *scraping* nama dan deskripsi makanan dari berbagai *website.* Keluarannya berupa data makanan yang kemudian masuk ke tahap praproses data. Pada tahap praproses data dilakukan *case folding*, penghilangan tanda baca dan simbol-simbol yang tidak terbaca, tokenisasi, serta *stopword removal*. Data yang sudah di praproses selanjutnya digunakan untuk membuat dua model yang berbeda, yaitu model NER dan model POS *Tagging*. Hasil model NER digunakan untuk klasifikasi label yang kemudian hasil klasifikasinya disimpan ke dalam *triple store*. Sedangkan hasil model POS *Tagging* digunakan dalam proses *Rule-based Matching.* Hasil R*ule-based Matching* kemudian juga ditambahkan ke dalam *triple store* untuk memperkaya korpus data makanan yang telah dibuat.

3.3 Kasus Penggunaan Sistem

Diagram kasus pengunaan dapat dilihat pada Gambar 3.3. Terdapat satu aktor yang terlibat dan berinteraksi langsung dengan sistem. Aktor yang berinteraksi dengan sistem yaitu pengguna yang diasumsikan tidak memahami bahasa pemrograman. Pengguna bisa dari berbagai profesi ataupun latar belakang yang berbeda-beda dengan rentang usia remaja sampai dewasa.

Selain aktor, pada diagram tersebut juga digambarkan dua kasus penggunaan, yaitu memasukkan data makanan dan mencari data makanan. Daftar kasus penggunaan sistem dapat dilihat pada Tabel 3.1.



Gambar 3.3 Diagram kasus penggunaan

Tabel 3.1 Daftar kasus penggunaan

Kode	Nama Kasus Penggunaan
UC01	Memasukkan data makanan
UC02	Mencari data makanan

3.3.1 Memasukkan Data Makanan

Pada kasus penggunaan ini, pengguna dapat memasukkan data makanan berupa nama dan deskripsi makanan yang akan dianotasi otomatis oleh sistem. Spesifikasi kasus penggunaan dapat dilihat pada Tabel 3.2.

Tabel 3.2 Spesifikasi kasus penggunaan UC01

Kode Kasus Penggunaan	UC01
Nama Kasus Penggunaan	Memasukkan Data Makanan
Aktor	Pengguna
Deskripsi	Pengguna dapat memasukkan data makanan berupa nama dan deskripsi makanan pada sistem
Relasi	*Directed Association*
Kondisi Awal	Data makanan belum tersimpan pada *triple store*
Kondisi Akhir	Data makanan tersimpan pada *triple store*
Alur Normal	
Aktor	Sistem
1. Pengguna memilih pilihan *input* data makanan	
	2. Sistem menampilkan isian *input* data
3. Pengguna memasukkan data makanan ke dalam isian *input* data. Data meliputi nama makanan dan deskripsi makanan.	
	4. Sistem memeriksa data yang dimasukkan oleh pengguna.

	A1. Data yang dimasukkan pengguna tidak lengkap. 5. Sistem melakukan proses anotasi otomatis terhadap deskripsi makanan sekaligus menyimpan data dan entitas penting hasil anotasi ke dalam *triple store*. 6. Sistem menampilkan hasil anotasi otomatis.
Alur Alternatif	
A1. Data yang dimasukkan pengguna tidak lengkap	
Aktor	Sistem
A1.2. Pengguna mendapatkan pesan eror. A1.3. Pengguna kembali ke alur nomor 3	A1.1. Sistem menampilkan pesan eror bahwa ada data yang belum lengkap atau belum diisi
Eksepsi	
-	

3.3.2 Mencari Data Makanan

Pada kasus penggunaan ini, pengguna dapat mencari data makanan berdasarkan kata kunci pencarian yang berupa nama, bahan, rasa, cara memasak makanan, atau gabungan dua atau lebih dari keempatnya. Spesifikasi kasus penggunaan dapat dilihat pada Tabel 3.3.

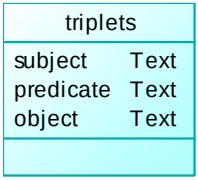
Tabel 3.3 Spesifikasi kasus penggunaan UC02

Kode Kasus Penggunaan	UC02
Nama Kasus Penggunaan	Mencari Data Makanan
Aktor	Pengguna
Deskripsi	Pengguna dapat mencari data makanan
Relasi	*Directed Association*
Kondisi Awal	Data makanan sudah tersimpan pada *triple store*
Kondisi Akhir	Data makanan ditampilkan pada sistem
Alur Normal	
Aktor	Sistem
1. Pengguna memilih pilihan cari makanan	
	2. Sistem menampilkan isian kata kunci pencarian
3. Pengguna memasukkan kata kunci pencarian. Kata kunci dapat berupa nama, bahan, rasa, cara memasak makanan, atau gabungan dua atau lebih dari keempatnya.	
	4. Sistem menampilkan hasil pencarian makanan yang sesuai dengan kata kunci.
5. Pengguna melihat hasil pencarian	
Alur Alternatif	
-	
Eksepsi	
-	

3.4 Analisis dan Perancangan Data

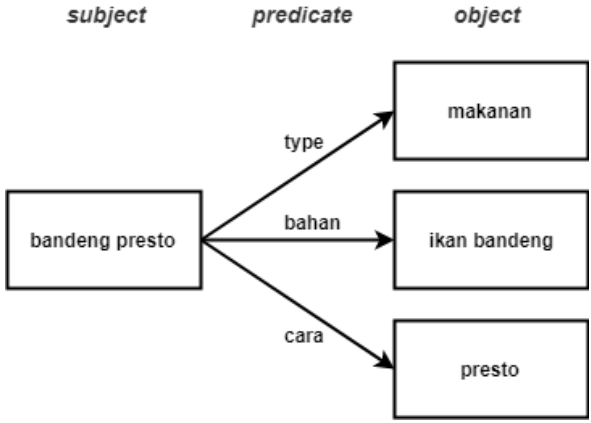
Pada subbab ini akan dijelaskan mengenai perancangan data yang digunakan pada pembuatan tugas akhir. Data yang digunakan merupakan nama makanan dan kalimat-kalimat deskripsi makanan yang diambil dari Wikipedia dan berbagai *blog* yang memuat aneka makanan khas di Indonesia, seperti Blog Tokopedia, GoTravelly, Tempat Wisata Indonesia, dan Indonesia Kaya. Deskripsi makanan tersebut mencakup bahan, rasa, dan cara memasak makanan. Pengambilan data dilakukan secara manual dan juga menggunakan *web crawlers* Octoparse [13]. Jumlah data yang digunakan sebanyak 1.535 kalimat deskripsi dari 330 makanan. Contoh data yang digunakan dapat dilihat pada Lampiran 1.

Data makanan disimpan ke dalam *triple store* Apache Jena Fuseki dalam bentuk pasangan *triplets* yang merupakan entitas data dengan susunan *subject-predicate-object*. Model data *triplets* ditunjukkan pada Gambar 3.4. Setiap makanan akan dimasukkan sebagai *instance* baru dalam *class* ‘makanan’. Selain itu, *instance* makanan tersebut juga memiliki atribut *property* berupa *annotation property* yang digunakan untuk menyimpan entitas bahan, cara, rasa, dan deskripsi makanan. Contoh pemodelan data *triplets* ditunjukkan oleh Gambar 3.5.



Gambar 3.4 Model data *triplets*

Pada Gambar 3.5 penulis mengambil salah satu makanan sebagai contoh, yaitu bandeng presto. Dalam pemodelan data *triplets*, bandeng presto merupakan *subject* yang memiliki banyak *predicate* yang terhubung dengan *object*. Misalnya pada gambar ditunjukkan bahwa bandeng presto memiliki *predicate type* dengan *object* makanan, memiliki *predicate* bahan dengan *object* ikan bandeng, dan memiliki *predicate* cara dengan *object* presto. Ketiga pasangan *triplets* tersebut dapat diartikan secara harfiah bahwa bandeng presto termasuk jenis makanan, yang memiliki bahan bandeng, dan dimasak dengan cara presto. Dari contoh pemodelan data *triplets* bandeng presto tersebut apabila disimpan ke *triple store* bentuk formatnya menjadi seperti yang ditunjukkan oleh Tabel 3.4.



Gambar 3.5 Contoh pemodelan data *triplets*

Tabel 3.4 Contoh format data *triplets* yang disimpan dalam *triple store*

	subject	***predicate***	***object***
1	ont:bandeng_presto	rdf:type	ont:makanan
2	ont:bandeng_presto	ont:bahan	“ikan bandeng”
3	ont:bandeng_presto	ont:cara	“presto”

3.5 Perancangan Proses

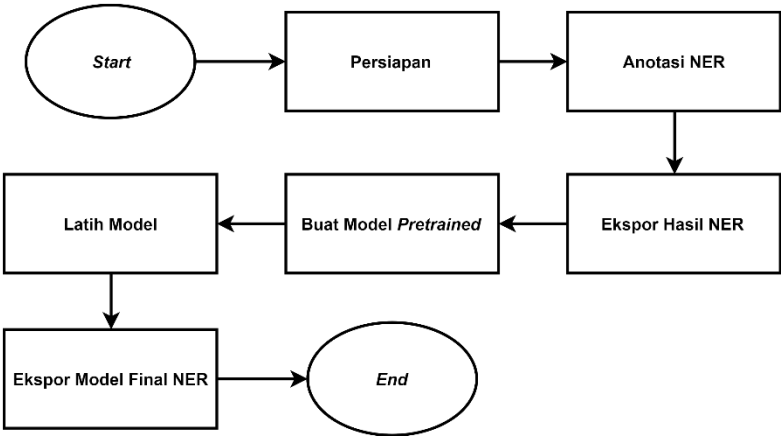
Perancangan proses merupakan penjelasan proses apa saja yang dilakukan untuk tugas akhir ini.

3.5.1 Pembuatan Model NER

NER digunakan untuk mendeteksi nama, bahan, rasa, dan cara memasak makanan yang ada dalam deskripsi makanan. Label yang digunakan dalam pembuatan model NER dapat dilihat pada Tabel 3.5.

Tabel 3.5 Label yang digunakan dalam model NER

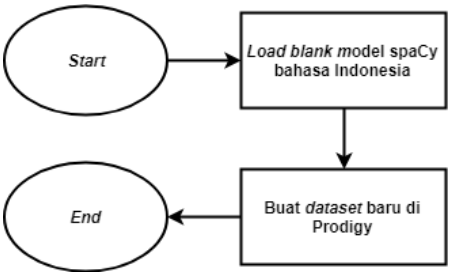
Nama Label	Keterangan
NAMA	Nama makanan
BAHAN	Bahan yang digunakan dalam proses pembuatan makanan
RASA	Rasa makanan, seperti: manis, asam, asin, gurih, pedas
CARA	Cara memasak makanan, seperti: goreng, rebus, kukus, panggang, dll.



Gambar 3.6 Diagram alir pembuatan model NER

Diagram alir pembuatan model NER dapat dilihat pada Gambar 3.6. Rincian masing-masing diagram pembuatan model NER dijelaskan pada poin-poin sebagai berikut:

1. Persiapan
   Pada tahap persiapan dilakukan *load blank* model bahasa Indonesia dari spaCy. Perlu menggunakan *blank* model, karena model NER untuk bahasa Indonesia belum tersedia di spaCy. Selain itu, pada tahap ini juga dilakukan pembuatan *dataset* baru pada Prodigy yang nantinya digunakan untuk menyimpan hasil anotasi.



Gambar 3.7 Diagram tahap persiapan

2. Anotasi NER
   Pada tahap ini dilakukan proses anotasi NER secara manual dengan melabeli setiap entitas nama, bahan, rasa, dan cara memasak makanan pada kalimat deskripsi makanan.
3. Ekspor Hasil NER
   Hasil anotasi NER diekspor dalam format JSON untuk selanjutnya digunakan dalam pembuatan model *pre-trained*.
4. Buat Model *Pretrained*
   Pembuatan model *pretrained* menggunakan hasil anotasi NER dari tahap sebelumnya.
5. Latih Model
   Model *pretrained* yang sudah dibuat harus dilatih terlebih dahulu menggunakan kalimat-kalimat deskripsi makanan yang

baru agar dapat lebih mengenali entitas-entitas penting yang ada dalam deskripsi makanan.

6. Ekspor Model Final NER
   Setelah dilatih, baru kemudian hasilnya diekspor menjadi model final NER yang siap digunakan.

Contoh rancangan *input* dan *output* NER dapat dilihat pada Tabel 3.6. Hasil dari NER kemudian disimpan sebagai anotasi sesuai dengan labelnya pada *triple store* Apache Jena Fuseki. Proses penyimpanan anotasi akan dijelaskan pada subbab 3.5.4.

Tabel 3.6 Contoh rancangan *input* dan *output* NER

Contoh *Input*	**Contoh *Output* Hasil NER**
Bandeng presto adalah makanan khas Indonesia yang berasal dari Kota Semarang, Jawa Tengah. Makanan ini dibuat dari ikan bandeng yang dibumbui dengan bawang putih, kunyit dan garam. Ikan bandeng ini kemudian dimasak pada alas daun pisang dengan cara presto.	[('bandeng presto', 'NAMA'), ('ikan bandeng', 'BAHAN'), ('bawang putih', 'BAHAN'), ('kunyit', 'BAHAN'), ('garam', 'BAHAN'), ('presto', 'CARA')]

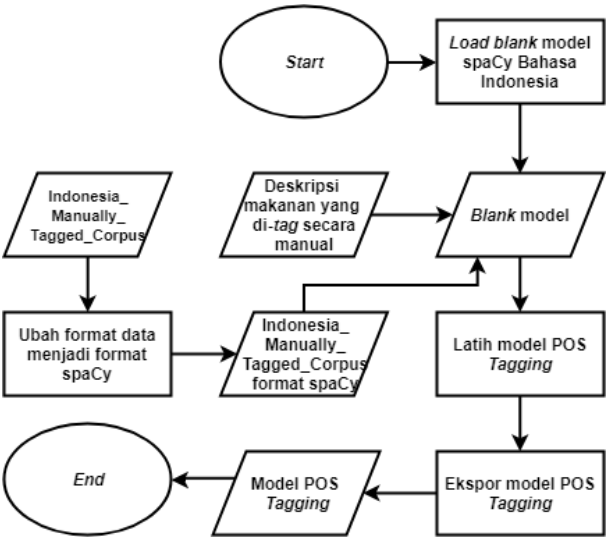
3.5.2 Pembuatan Model POS *Tagging*

POS *Tagging* digunakan untuk mengidentifikasi kelas kata yang ada dalam kalimat deskripsi makanan. Model POS *Tagging* dibuat menggunakan data Indonesian_Manually_Tagged_Corpus yang diambil dari GitHub famrashel [14]. Format data tersebut terlebih dahulu disesuaikan dengan format POS *Tagging* pada spaCy. Selain itu, penulis juga menambahkan 50 data deskripsi makanan yang telah di-*tag* secara manual sesuai dengan *part-of-speech* setiap katanya. Kedua data tersebut kemudian digunakan

sebagai data latih untuk membuat model POS *Tagging*. Diagram alir pembuatan model POS *Tagging* dapat dilihat pada Gambar 3.8. Daftar *tag* yang digunakan [15] dapat dilihat pada Tabel 3.7. Contoh rancangan *input* dan *output* POS *Tagging* dapat dilihat pada Tabel 3.8. Hasil dari POS *Tagging* selanjutnya digunakan dalam proses *Rule-based Matching* yang akan dijelaskan pada subbab 3.5.3.

Tabel 3.7 Daftar *tag*

Tag	**Keterangan**	**Contoh**
ADJ	*adjective* (kata sifat)	panjang, hitam, manis, muda
ADP	*adposition* (kata depan)	di, ke, dalam, pada
ADV	*adverb* (kata keterangan)	sangat, hanya, segera
AUX	*auxiliary verb* (verba bantu)	boleh, harus, sudah
CONJ	*coordinating conjuction* (konjungsi koordinatif)	dan, tetapi, atau
DET	*determiner* (artikel)	para, sebuah
INTJ	*interjection* (interjeksi)	ayo, mari, wah
NOUN	*noun* (nomina/kata benda)	ayam, ikan, sayur
NUM	*numeral* (numeralia)	satu, kedua, banyak
PART	*particle* (partikel)	pun, -lah, -kah
PRON	*pronoun* (pronomina)	anda, kita
PROPN	*proper noun*	Surabaya, Jawa Timur, Indonesia
PUNCT	*punctuation* (tanda baca)	“…”, ?, .
SCONJ	*subordinating conjunction* (konjungtor subordinatif)	jika, supaya, maka
SYM	*symbol* (simbol)	+, %
VERB	*verb* (verba/kata kerja)	menggoreng, rebus, dipanggang
X	*other* (belum diketahui kategorinya secara pasti)	



Gambar 3.8 Diagram alir pembuatan model POS *Tagging*

Tabel 3.8 Contoh rancangan *input* dan *output* POS *Tagging*

Contoh *Input*	**Contoh *Output* Hasil POS *Tagging***
Bandeng presto adalah makanan khas Indonesia yang berasal dari Kota Semarang, Jawa Tengah. Makanan ini dibuat dari ikan bandeng yang dibumbui dengan bawang putih, kunyit dan garam. Ikan bandeng ini kemudian dimasak pada alas daun pisang dengan cara presto.	[('bandeng', 'NOUN'), ('presto', 'VERB'), ('adalah', 'VERB'), ('makanan', 'NOUN'), ('khas', 'ADJ'), ('indonesia', 'PROPN'), ('yang', 'SCONJ'), ('berasal', 'VERB'), ('dari', 'ADP'), ('kota', 'PROPN'), ('semarang', 'PROPN'), (',', 'PUNCT'), ('jawa', 'PROPN'), ('tengah', 'PROPN'), ('.', 'PUNCT'), ('makanan', 'NOUN'), ('ini', 'PRON'), ('dibuat', 'VERB'), ('dari', 'ADP'), ('ikan', 'NOUN'), ('bandeng', 'NOUN'), ('yang', 'SCONJ'), ('dibumbui', 'VERB'), ('dengan',

Contoh *Input*	Contoh *Output* Hasil POS *Tagging*
	'ADP'), ('bawang', 'NOUN'), ('putih', 'NOUN'), (',', 'PUNCT'), ('kunyit', 'NOUN'), ('dan', 'CONJ'), ('garam', 'NOUN'), ('.', 'PUNCT'), ('ikan', 'NOUN'), ('bandeng', 'NOUN'), ('ini', 'PRON'), ('kemudian', 'ADV'), ('dimasak', 'VERB'), ('pada', 'ADP'), ('alas', 'NOUN'), ('daun', 'NOUN'), ('pisang', 'NOUN'), ('dengan', 'ADP'), ('cara', 'NOUN'), ('presto', 'VERB'), ('.', 'PUNCT')]

3.5.3 Pembuatan *Rule*

Rule digunakan dalam proses *Rule-based Matching*, yaitu pencocokkan kata atau frasa tertentu sesuai dengan *rule* yang telah dibuat. Proses ini nantinya digunakan untuk mendapatkan kata atau frasa penting yang belum terdeteksi oleh NER. Pembuatan *rule* mengacu pada hasil POS *Tagging* yang telah dilakukan sebelumnya. Daftar *rule* yang digunakan dapat dilihat pada Tabel 3.9. Contoh rancangan *input* dan *output Rule-based Matching* dapat dilihat pada Tabel 3.10. Hasil dari *Rule-based Matching* digunakan untuk memperkaya korpus data makanan terutama untuk memperkaya entitas bahan yang mungkin belum terdeteksi ketika menggunakan NER.

Tabel 3.9 Daftar *rule*

Rule	**Fungsi**
NOUN + NOUN	Mendeteksi entitas bahan

Tabel 3.10 Contoh rancangan *input* dan *output Rule-based Matching*

Contoh *Input*	**Contoh *Output* Hasil *Rule-based Matching***
Bandeng presto adalah makanan khas Indonesia yang berasal dari Kota Semarang, Jawa Tengah. Makanan ini dibuat dari ikan bandeng yang dibumbui dengan bawang putih, kunyit dan garam. Ikan bandeng ini kemudian dimasak pada alas daun pisang dengan cara presto.	['bandeng', 'ikan', 'ikan bandeng', 'bawang', 'bawang putih', 'kunyit', 'garam']

3.5.4 Penyimpanan Anotasi

Hasil NER disimpan sebagai anotasi pada *triple store* Apache Jena Fuseki. Cara penyimpanan anotasi dilakukan dengan mengubah format hasil NER menjadi sintaks SPARQL dengan menggunakan *script* Python. Sintaks SPARQL tersebut kemudian digunakan sebagai *query* pada Apache Jena Fuseki. Contoh hasil NER dan sintaks SPARQL yang merupakan konversi dari hasil NER ditunjukkan pada Tabel 3.11.

Tabel 3.11 Contoh rancangan penyimpanan anotasi

Contoh Hasil NER	**Contoh Sintaks SPARQL**
[('bandeng presto', 'NAMA'), ('ikan bandeng', 'BAHAN'), ('bawang putih', 'BAHAN'), ('kunyit',	INSERT DATA { ont:bandeng presto ont:bahan "ikan bandeng"}; INSERT DATA { ont:bandeng presto ont:bahan "bawang putih"}; INSERT DATA { ont:bandeng presto ont:bahan "kunyit"};

Contoh Hasil NER	Contoh Sintaks SPARQL
'BAHAN'), ('garam', 'BAHAN'), ('presto', 'CARA')]	INSERT DATA { ont:bandeng presto ont:bahan "garam"}; INSERT DATA { ont:bandeng presto ont:cara "presto"};

3.6 Perancangan Antarmuka Sistem

Antarmuka sistem pada tugas akhir ini digunakan untuk menampilkan demo simulasi hasil NER, POS *Tagging*, dan *Rule-based Matching* dengan melakukan *input* nama dan deskripsi makanan. Selain itu, antarmuka sistem juga dapat menampilkan hasil *query* atau pencarian makanan sesuai dengan kata kunci pencarian yang dimasukkan oleh pengguna.

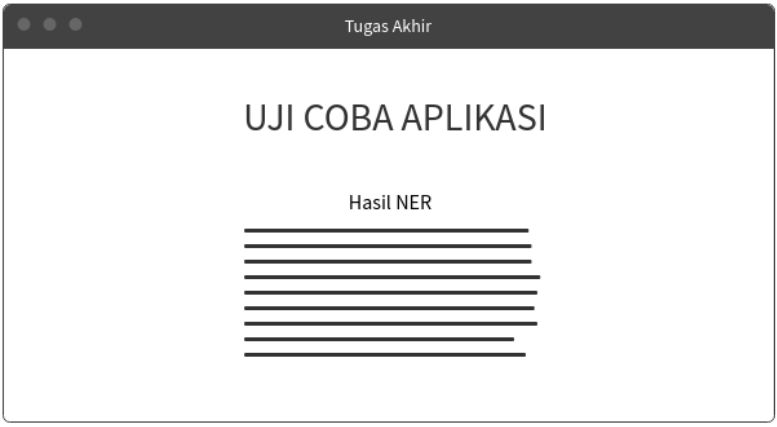
Gambar 3.9 Rancangan antarmuka halaman beranda aplikasi

Rancangan halaman depan atau beranda aplikasi seperti ditunjukkan pada Gambar 3.9, terdapat dua tombol yang memiliki fungsi berbeda, yang pertama untuk *input* data makanan dan yang

kedua untuk mencari makanan. Apabila tombol *input* data makanan diklik, maka akan berpindah ke halaman *input* data makanan. Pada halaman *input* data makanan, pengguna dapat memasukkan nama makanan dan deskripsi makanan pada *form* seperti ditunjukkan oleh Gambar 3.10. Setelah memasukkan nama dan deskripsi makanan, pengguna harus menekan tombol Submit agar sistem memproses NER dan menampilkan hasilnya seperti ditunjukkan oleh Gambar 3.11.

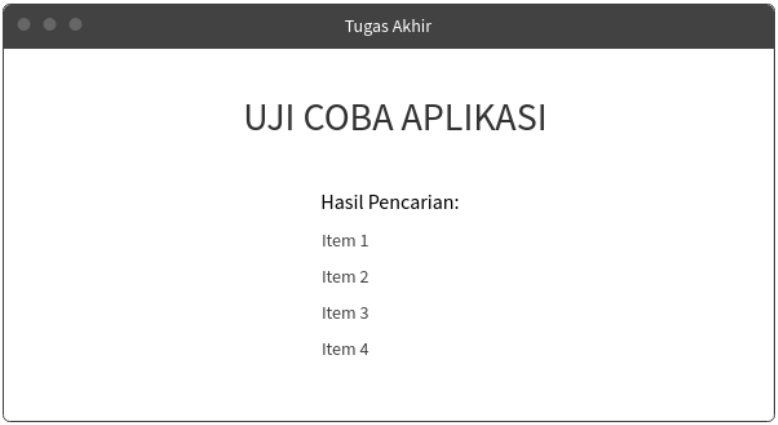
Gambar 3.10 Rancangan antarmuka halaman *input* data makanan

Sedangkan apabila pada halaman beranda aplikasi, pengguna memilih tombol cari makanan, maka akan berpindah ke halaman cari makanan. Pada halaman cari makanan, pengguna dapat memasukkan kata kunci pencarian pada *form* seperti ditunjukkan pada Gambar 3.12. Setelah memasukkan kata kunci sesuai dengan keinginan, pengguna harus menekan tombol Submit agar sistem memproses pencarian. Sistem akan menampilkan hasil pencarian makanan yang sesuai dengan kata kunci seperti ditunjukkan pada Gambar 3.13.



Gambar 3.11 Rancangan antarmuka halaman hasil NER

Gambar 3.12 Rancangan antarmuka halaman cari makanan



Gambar 3.13 Rancangan antarmuka halaman hasil pencarian makanan

3.7 Evaluasi dan Uji Coba

Pada tahap evaluasi dan uji coba dilakukan evaluasi untuk model NER, uji coba *input* data, uji coba *query* label, dan uji coba bentuk anotasi yang sesuai dengan standar. Evaluasi model NER dilakukan penulis secara manual dengan menghitung nilai *precision*, *recall*, dan *f-measure* untuk setiap label NER. Sedangkan untuk uji coba *input* data, *query* label, dan bentuk anotasi dilakukan penulis secara manual dengan cara uji coba langsung pada sistem.

BAB IV
IMPLEMENTASI

Pada bab ini akan dijelaskan implementasi yang dilakukan selama tugas akhir. Bab ini juga akan merinci perangkat serta *tools* yang digunakan pada tugas akhir ini beserta langkah-langkah pengerjaannya.

4.1 Lingkungan Implementasi

Pada tugas akhir ini digunakan beberapa perangkat serta *tools* yang membantu dalam pengerjaan tugas akhir. Perangkat keras dan sistem operasi yang digunakan selama tugas akhir adalah:

- Processor Intel® Core™ i7-4720HQ CPU @ 2.60GHz
- Installed RAM 12.0 GB (11.9 GB usable)
- System Type 64-bit operating system, x64-based processor
- Windows Edition Windows 8.1 Pro

Sedangkan untuk *tools* yang digunakan selama tugas akhir diuraikan pada Tabel 4.1.

Tabel 4.1 *Tools* yang digunakan pada tugas akhir

No.	*Tools*	Keterangan
1	Python	Bahasa Python digunakan untuk menangani *task Natural Language Processing* (NLP).
2	spaCy	Pustaka untuk melakukan NER, POS *Tagging*, dan *Rule-based Matching*
3	Octoparse	Aplikasi untuk *scraping* halaman web.
4	Flask	Kerangka kerja aplikasi web Python.
5	Apache Jena Fuseki	*Triple store* untuk menyimpan data

No.	*Tools*	Keterangan
6	Jupyter Notebook, Sublime Text 3, Visual Studio Code	Aplikasi yang digunakan untuk penulisan program.
7	Ms. Excel dan Ms. Word	Aplikasi untuk mengolah data.

4.2 Persiapan dan Pengambilan Data

Data deskripsi makanan dari Wikipedia dan berbagai *blog* yang memuat aneka makanan khas di Indonesia diambil dengan proses *scraping* menggunakan *tools* Octoparse dan juga secara manual.

4.3 Implementasi Proses

Implementasi proses dilakukan berdasarkan perancangan proses yang dijelaskan pada bab Analisis dan Perancangan Sistem.

4.3.1 Implementasi Pembuatan Model NER

Implementasi pembuatan model NER dilakukan dengan menggunakan bantuan alat anotasi Prodigy dan pustaka spaCy.

4.3.1.1 Load Blank Model

Perlu menggunakan *blank* model, karena model NER untuk bahasa Indonesia belum tersedia di spaCy. Cara me-*load blank* model dapat dilihat pada Kode Sumber 4.1. Pada kode sumber tersebut, *blank* model di-*load* pada direktori ‘C:\Users\ASUS CORNER\model’.

```
py -c "import spacy;
spacy.blank('id').to_disk(r'C:\Users\ASUS
CORNER\model')"
```

Kode Sumber 4.1 *Load Blank* Model

4.3.1.2 Pembuatan *Dataset* Baru di Prodigy

Dataset memungkinkan pengguna untuk mengelompokkan anotasi. Cara membuat *dataset* baru di Prodigy dapat dilihat pada Kode Sumber 4.2. Pada kode sumber tersebut, *dataset* baru diberi nama ‘deskripsi_makanan’.

```
py -m prodigy dataset deskripsi_makanan
```

Kode Sumber 4.2 Pembuatan *dataset* baru di Prodigy

4.3.1.3 Anotasi NER Manual Menggunakan Prodigy

Anotasi NER secara manual dilakukan untuk memberi label secara manual pada entitas kata yang dianggap penting. Cara menganotasi NER manual pada Prodigy dengan menggunakan perintah *ner.manual* seperti ditunjukkan pada Kode Sumber 4.3. Pada kode sumber tersebut, *file* data yang akan dianotasi memiliki nama ‘data_train.txt’ dan label yang digunakan ada 4 macam, yaitu ‘NAMA’, ‘BAHAN’, ‘RASA’, dan ‘CARA’. Penjelasan masing-masing label dapat dilihat pada subbab 3.5.1. Contoh data yang akan dianotasi ditunjukkan oleh Gambar 4.1.

```
py -m prodigy ner.manual deskripsi_makanan
“C:\Users\ASUS CORNER\model" data_train.txt --label
"NAMA, BAHAN, RASA, CARA"
```

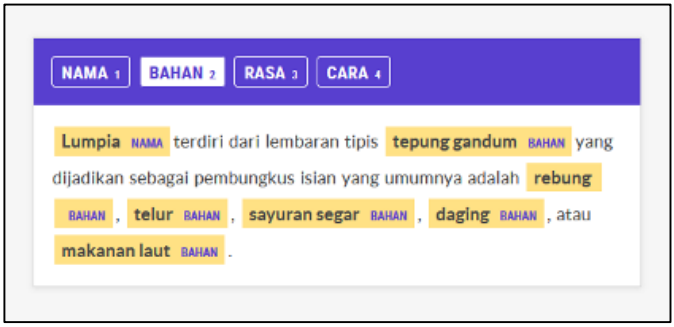
Kode Sumber 4.3 Anotasi NER manual menggunakan *ner.teach*

```
1   Bandeng presto adalah makanan khas Indonesia yang berasal dari Kota
    Semarang, Jawa Tengah.
2   Makanan ini dibuat dari ikan bandeng yang dibumbui dengan bawang putih,
    kunyit dan garam.
3   Ikan bandeng ini kemudian dimasak pada alas daun pisang dengan cara
    presto.
4   Tahu campur adalah salah satu makanan khas Jawa Timur.
5   Tahu campur terdiri dari sop daging sapi kenyal, tahu goreng, perkedel
    singkong, taoge segar, selada air segar, mi kuning, dan kerupuk udang.
6   Semua ini kemudian dicampurkan ke bumbu petis, bawang goreng, dan sambal.
```

Gambar 4.1 Contoh data yang akan dianotasi

Selanjutnya *web server* Prodigy akan menyala dan penulis dapat melakukan anotasi NER secara manual melalui Prodigy pada *browser*. Tampilan Prodigy sebelum dilakukan anotasi (tampilan awal) dapat dilihat pada Gambar 4.2 dan tampilan saat penulis melakukan anotasi dapat dilihat pada Gambar 4.3. Penulis melakukan anotasi terhadap 533 kalimat deskripsi makanan.

Gambar 4.2 Tampilan Prodigy sebelum dilakukan anotasi NER



Gambar 4.3 Tampilan Prodigy saat dilakukan anotasi NER

4.3.1.4 Mengekspor Hasil Anotasi NER

Hasil anotasi NER diekspor dalam format JSONL agar selanjutnya dapat digunakan untuk pembuatan model NER. Dari proses anotasi yang telah dilakukan sebelumnya, data yang

terekspor hanyalah data yang bernilai *accepted*. Sedangkan untuk data yang bernilai *rejected* dan *ignore* tidak akan terekspor. Hal tersebut menyebabkan jumlah data hasil ekspor mungkin tidak sama dengan jumlah data awal. Dari data awal sebanyak 543 kalimat, jumlah data yang berhasil diekspor sebanyak 491 kalimat deskripsi makanan.

Cara mengekspor hasil NER dari Prodigy dengan menggunakan perintah *ner.gold-to-spacy* seperti ditunjukkan pada Kode Sumber 4.4. Berdasarkan kode sumber tersebut, hasil anotasi NER akan disimpan pada direktori "C:\Users\ASUS CORNER\ner\hasil_anotasi.jsonl". Contoh hasil anotasi NER ditunjukkan oleh Gambar 4.4.

```
py -m prodigy ner.gold-to-spacy  deskripsi_makanan
"C:\Users\ASUS CORNER\ner\hasil_anotasi.jsonl"
```

Kode Sumber 4.4 Mengekspor hasil anotasi NER


```
1  ["Bandeng presto adalah makanan khas Indonesia yang berasal dari Kota
   Semarang, Jawa Tengah.",{"entities":[[0,14,"NAMA"]]}]
2  ["Makanan ini dibuat dari ikan bandeng yang dibumbui dengan bawang putih,
   kunyit dan garam.",{"entities":[[24,36,"BAHAN"],[58,70,"BAHAN"],[72,78,
   "BAHAN"],[83,88,"BAHAN"]]}]
3  ["Ikan bandeng ini kemudian dimasak pada alas daun pisang dengan cara
   presto.",{"entities":[[0,12,"BAHAN"],[68,74,"CARA"]]}]
4  ["Tahu campur adalah salah satu makanan khas Jawa Timur.",{"entities":[[0,
   11,"NAMA"]]}]
5  ["Tahu campur terdiri dari sop daging sapi kenyal, tahu goreng, perkedel
   singkong, taoge segar, selada air segar, mi kuning, dan kerupuk udang.",
   {"entities":[[0,11,"NAMA"],[25,47,"BAHAN"],[49,60,"BAHAN"],[62,79,"BAHAN"]
   ,[81,92,"BAHAN"],[94,110,"BAHAN"],[112,121,"BAHAN"],[127,140,"BAHAN"]]}]
6  ["Semua ini kemudian dicampurkan ke bumbu petis, bawang goreng, dan
   sambal.",{"entities":[[34,45,"BAHAN"],[47,60,"BAHAN"],[66,72,"BAHAN"]]}]
```


Gambar 4.4 Contoh hasil anotasi NER

4.3.1.5 Pembuatan Model *Pre-trained* NER

Pembuatan model NER dilakukan dengan menggunakan kode program yang diambil dari GitHub ManivannanMurugavel [16]. Hasil anotasi NER digunakan sebagai data latih untuk

membuat model. Cara membuat model NER dapat dilihat pada Kode Sumber 4.5. Pada kode sumber tersebut, data *train* yang digunakan adalah kalimat-kalimat hasil anotasi NER. Awalnya *load blank* model bahasa Indonesia terlebih dahulu. Kemudian menambahkan label sesuai dengan entitas NER. Setiap entitas NER selanjutnya akan di-*train* dengan iterasi sebanyak jumlah data. Hasil *training* berupa model NER yang akan disimpan ke *disk*. Di akhir kode sumber, model NER diuji coba dengan masukan teks baru berupa deskripsi makanan untuk mengetahui apakah model yang dibuat sudah benar dapat mengenali entitas nama, bahan, rasa, dan cara memasak makanan.

```
1.  import spacy, random
2.
3.  TRAIN_DATA = []
4.
5.  def train_spacy(data,iterations):
6.      TRAIN_DATA = data
7.      nlp = spacy.blank('id')  # create blank Langu
    age class
8.      # create the built-
    in pipeline components and add them to the pipeli
    ne
9.      # nlp.create_pipe works for built-
    ins that are registered with spaCy
10.     if 'ner' not in nlp.pipe_names:
11.         ner = nlp.create_pipe('ner')
12.         nlp.add_pipe(ner, last=True)
13.
14.     # add labels
15.     for _, annotations in TRAIN_DATA:
16.          for ent in annotations.get('entities'):

17.             ner.add_label(ent[2])
18.
19.     # get names of other pipes to disable them du
    ring training
20.     other_pipes = [pipe for pipe in nlp.pipe_name
    s if pipe != 'ner']
```

```
21.     with nlp.disable_pipes(*other_pipes):  # only
    train NER
22.         optimizer = nlp.begin_training()
23.         for itn in range(iterations):
24.             print("Starting iteration " + str(itn
    ))
25.             random.shuffle(TRAIN_DATA)
26.             losses = {}
27.             for text, annotations in TRAIN_DATA:

28.                 nlp.update(
29.                     [text],  # batch of texts
30.                     [annotations],  # batch of an
    notations
31.                     drop=0.2,  # dropout - make i
    t harder to memorise data
32.                     sgd=optimizer,  # callable to
    update weights
33.                     losses=losses)
34.             print(losses)
35.     return nlp
36.
37. prdnlp = train_spacy(TRAIN_DATA, 491)
38.
39. # Save our trained Model
40. modelfile = "model_ner"
41. prdnlp.to_disk(modelfile)
42.
43. #Test your text
44. test_text = "Kue putu adalah jenis makan tradisio
    nal nusantara yang berupa kue dengan isian gula j
    awa, dibalut dengan parutan kelapa, dan tepung be
    ras butiran kasar. Kue ini dikukus dengan diletak
    kan di dalam tabung bambu yang sedikit dipadatkan
    ."
45. print()
46. print("Test text: \n" + test_text)
47. print()
48. print("Result: ")
49. doc = prdnlp(test_text)
50. for ent in doc.ents:
```

```
    print(ent.text, ent.start_char, ent.end_char, ent.label_)
```

Kode Sumber 4.5 Pembuatan model *pre-trained* NER

4.3.1.6 Melatih Model NER

Model NER perlu dilatih agar dapat lebih mengenali entitas-entitas penting yang ada dalam deskripsi makanan. Proses tersebut dilakukan di Prodigy dengan menggunakan dua perintah, *ner.teach* dan *ner.make-gold*. Ketika perintah *ner.teach* dijalankan, Prodigy akan mengajukan pernyataan yang merupakan data masukan yang teranotasi secara acak berdasarkan model NER yang telah dibuat. Kemudian pengguna tinggal mengecek apakah anotasi yang diberikan sudah benar atau belum. Perintah *ner.teach* ditunjukkan oleh Kode Sumber 4.6 dan tampilan Prodigy saat *ner.teach* dijalankan dapat dilihat pada Gambar 4.5. Sedangkan ketika perintah *ner.make-gold* dijalankan, model NER yang telah dibuat akan digunakan untuk memprediksi entitas yang terkandung dalam teks dan Prodigy akan menampilkan anotasinya. Pengguna dapat mengoreksi saran anotasi yang diberikan oleh Prodigy dengan menghapus atau memperbaiki anotasi tersebut. Perintah *ner.make-gold* ditunjukkan oleh Kode Sumber 4.7 dan tampilan Prodigy saat *ner.make-gold* dijalankan dapat dilihat pada Gambar 4.6. Pada kode sumber tersebut, *dataset* yang digunakan memiliki nama ‘dataset_ner’ dan model yang digunakan terletak pada direktori "C:\Users\ASUS CORNER\model_ner". Ketika menjalankan *ner.teach*, penulis menggunakan *file* data dengan nama ‘data_training.txt’, sedangkan ketika menjalankan *ner.make-gold* menggunakan *file* data dengan nama ‘data_training_copy.txt’ yang merupakan duplikat dari *file* ‘data_training.txt’. Label yang digunakan saat *ner.make-gold* ada 4 macam, yaitu ‘NAMA’, ‘BAHAN’, ‘RASA’, dan ‘CARA’. Penulis melakukan anotasi sebanyak 619 kalimat deskripsi makanan saat menjalankan

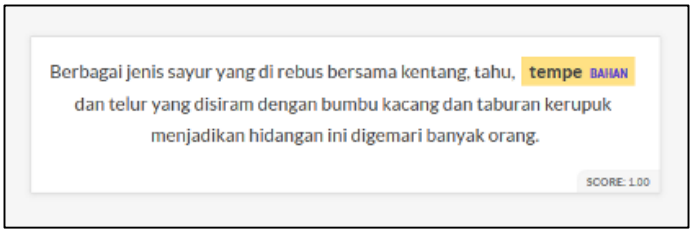
masing-masing perintah *ner.teach* dan *ner.make-gold*. Contoh data latih dapat dilihat pada Gambar 4.7.

```
py -m prodigy ner.teach dataset_ner "C:\Users\ASUS
CORNER\model_ner" data_training.txt
```

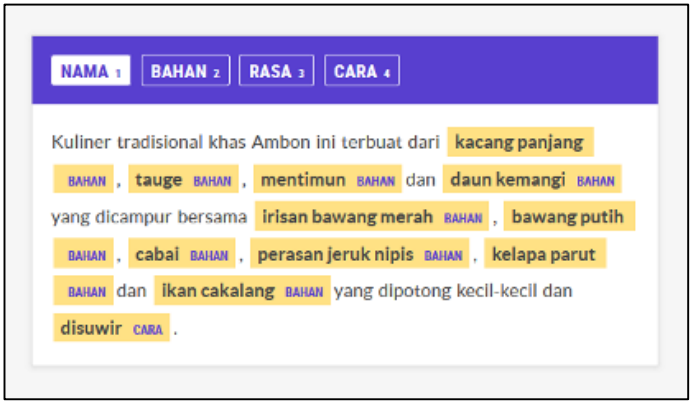
Kode Sumber 4.6 Melatih model NER menggunakan *ner.teach*

```
py -m prodigy ner.make-gold dataset_ner "C:\Users\ASUS
CORNER\model_ner" data_training_copy.txt --label "NAMA,
BAHAN, RASA, CARA"
```

Kode Sumber 4.7 Melatih model NER menggunakan *ner.make-gold*



Gambar 4.5 Tampilan Prodigy saat *ner.teach* dijalankan



Gambar 4.6 Tampilan Prodigy saat *ner.make-gold* dijalankan

```
1   Kohu-kohu merupakan makanan khas Ambon yang mirip dengan urap.
2   Kuliner tradisional khas Ambon ini terbuat dari kacang panjang, tauge,
    mentimun dan daun kemangi yang dicampur bersama irisan bawang merah,
    bawang putih, cabai, perasan jeruk nipis, kelapa parut dan ikan cakalang
    yang dipotong kecil-kecil dan disuwir.
3   Ikan cakalang juga dapat diganti dengan ikan tongkol atau ikan teri basah.
4   Kue cucur adalah salah satu kue tradisional khas Betawi yang sering dan
    mudah kita jumpai di pasar tradisional maupun di toko kue.
5   Kue cucur ini rasanya manis dengan tekstur lembut di bagian tengahnya,
    serta renyah di bagian pinggirnya.
6   Kue berbahan dasar beras dan gula merah ini dimasak dengan cara digoreng.
```

Gambar 4.7 Contoh data latih NER

4.3.1.7 Mengekspor Model NER

Hasil pelatihan model NER perlu diekspor dengan menggunakan perintah *ner.batch-train* seperti ditunjukkan pada Kode Sumber 4.8. Pada kode sumber tersebut, hasil ekspor model NER akan disimpan pada direktori "C:\Users\ASUS CORNER\model_ner_output". Prodigy akan mengekspor hasil terbaik ke direktori *output*, dan menyertakan file dengan format JSONL dari contoh pelatihan dan evaluasi. Secara *default*, 50% dipisahkan untuk *dataset* di bawah 1000, dan 20% untuk *dataset* lebih dari 1000. Namun, pengguna juga dapat menentukan sendiri persentase untuk evaluasi.

```
py -m prodigy ner.batch-train dataset_ner "C:\Users\ASUS
CORNER\model_ner" --output "C:\Users\ASUS
CORNER\model_ner_output"
```

Kode Sumber 4.8 Mengekspor model NER

4.3.2 Implementasi Pembuatan Model POS *Tagging*

Implementasi pembuatan POS *Tagging* dilakukan dengan menggunakan bantuan pustaka spaCy. Data latih diambil dari Indonesian_Manually_Tagged_Corpus dan deskripsi makanan yang telah di-*tag* secara manual sesuai dengan *part-of-speech* setiap katanya. Contoh data latih ditunjukkan oleh Gambar 4.8. Cara implementasi pembuatan POS *Tagging* dapat dilihat pada

Kode Sumber 4.9. Pada kode sumber tersebut, terlebih dahulu dilakukan konversi format *tag* agar sesuai dengan format yang dimiliki spaCy. Selanjutnya *load* blank model bahasa Indonesia dan *load* data *train*. Kemudian setiap *tag part-of-speech* akan di-*train*. Hasil *training* berupa model POS *Tagging* yang akan disimpan ke *disk*. Di akhir kode sumber, model POS *Tagging* diuji coba dengan masukan teks baru untuk mengetahui apakah model yang dibuat sudah benar dapat mengenali setiap kelas kata dalam kalimat.

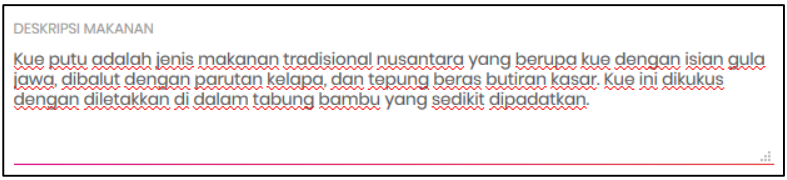
```
1   ('Bandeng presto adalah makanan khas Indonesia yang berasal dari Kota
    Semarang , Jawa Tengah .', {'tags': ['NN', 'NN', 'VB', 'NN', 'JJ', 'NNP',
    'SC', 'VB', 'IN', 'NNP', 'NNP', 'Z', 'NNP', 'NNP', 'Z']}),
2   ('Makanan ini dibuat dari ikan bandeng yang dibumbui dengan bawang putih ,
     kunyit dan garam .', {'tags': ['NN', 'PR', 'VB', 'IN', 'NN', 'NN', 'SC',
    'VB', 'IN', 'NN', 'NN', 'Z', 'NN', 'CC', 'NN', 'Z']}),
3   ('Ikan bandeng ini kemudian dimasak pada alas daun pisang dengan cara
    presto .', {'tags': ['NN', 'NN', 'PR', 'RB', 'VB', 'IN', 'NN', 'NN', 'NN',
     'IN', 'NN', 'NN', 'Z']}),
4   ('Tahu campur adalah salah satu makanan khas Jawa Timur .', {'tags':
    ['NN', 'VB', 'VB', 'CD', 'CD', 'NN', 'NN', 'NNP', 'NNP', 'Z']}),
5   ('Tahu campur terdiri dari sop daging sapi kenyal , tahu goreng ,
    perkedel singkong , taoge segar , selada air segar , mi kuning , dan
    kerupuk udang .', {'tags': ['NN', 'VB', 'VB', 'IN', 'NN', 'NN', 'NN',
    'JJ', 'Z', 'NN', 'VB', 'Z', 'NN', 'NN', 'Z', 'NN', 'JJ', 'Z', 'NN', 'NN',
    'JJ', 'Z', 'NN', 'NN', 'Z', 'CC', 'NN', 'NN', 'Z']}),
6   ('Semua ini kemudian dicampurkan ke bumbu petis , bawang goreng , dan
    sambal .', {'tags': ['CD', 'PR', 'RB', 'VB', 'IN', 'NN', 'NN', 'Z', 'NN',
    'VB', 'Z', 'CC', 'NN', 'Z']}),
```

Gambar 4.8 Contoh data latih POS *Tagging*

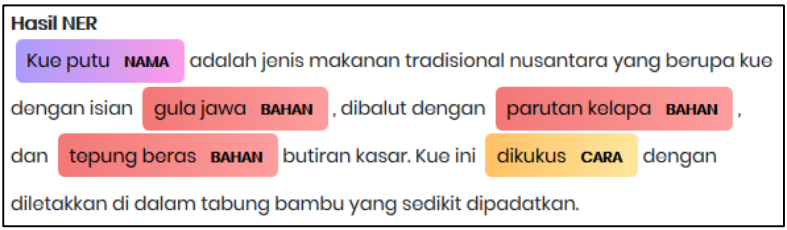
4.3.3 Implementasi NER

NER digunakan untuk mendeteksi nama, bahan, rasa, dan cara memasak makanan yang ada dalam deskripsi makanan. Implementasi NER pada sistem dengan menggunakan Flask dapat dilihat pada Kode Sumber 4.10. Pada kode sumber tersebut, penulis mengambil *value* deskripsi makanan dari *form* deskripsi dengan menggunakan *method request*. Kemudian *load* model NER yang sudah dibuat sebelumnya dan dilakukan deteksi NER pada kalimat deskripsi makanan dengan menggunakan model NER

tersebut. Hasil NER kemudian divisualisasikan dengan menggunakan bantuan pustaka displaCy agar terlihat lebih menarik dan mudah dipahami. Contoh data *input* ditunjukkan oleh Gambar 4.9 dan contoh *output* hasil NER ditunjukkan oleh Gambar 4.10.



Gambar 4.9 Contoh data *input*



Gambar 4.10 Contoh *output* hasil NER

```
1. from __future__ import unicode_literals, print_fu
   nction
2. import plac, random, spacy, pickle
3. from pathlib import Path
4. from spacy.util import minibatch, compounding
5.
6. #Convert tag to common POS format
7. TAG_MAP = { "CC": {"pos": "CONJ"}, "CD": {"pos":
   "NUM"}, "DT": {"pos": "DET"}, "FW": {"pos": "X"},
   "IN": {"pos": "ADP"}, "JJ": {"pos": "ADJ"}, "MD":
   {"pos": "AUX"}, "NEG": {"pos": "X"}, "NN": {"pos"
   : "NOUN"}, "NND": {"pos": "NOUN"}, "NNP": {"pos":
   "PROPN"}, "OD": {"pos": "NUM"}, "PR": {"pos": "PR
   ON"}, "PRP": {"pos": "PRON"}, "RB": {"pos": "ADV"
   }, "RP": {"pos": "PART"}, "SC": {"pos": "SCONJ"},
   "SYM": {"pos": "SYM"}, "UH": {"pos": "INTJ"}, "VB
```

```
    ": {"pos": "VERB"}, "WH": {"pos": "X"}, "XX": {"p
    os": "X"}, "Z": {"pos": "PUNCT"} }
8.
9.  cv = open("indonesian-manually-
    tagged.pkl","rb")
10. TRAIN_DATA = pickle.load(cv)
11. print(TRAIN_DATA)
12. print("Let's train %d strings!"%(len(TRAIN_DATA))
    )
13.
14. @plac.annotations(
15.     lang=("ISO Code of language to use", "option"
    , "l", str),
16.     output_dir=("Optional output directory", "opt
    ion", "o", Path),
17.     n_iter=("Number of training iterations", "opt
    ion", "n", int),
18. )
19. def main(lang="id", output_dir=None, n_iter=25):

20.     nlp = spacy.blank(lang)
21.     # add the tagger to the pipeline
22.     # nlp.create_pipe works for built-
    ins that are registered with spaCy
23.     tagger = nlp.create_pipe("tagger")
24.     # Add the tags. This needs to be done before
    you start training.
25.     for tag, values in TAG_MAP.items():
26.         tagger.add_label(tag, values)
27.     nlp.add_pipe(tagger)
28.
29.     optimizer = nlp.begin_training()
30.     for i in range(n_iter):
31.         random.shuffle(TRAIN_DATA)
32.         losses = {}
33.         # batch up the examples using spaCy's min
    ibatch
34.         batches = minibatch(TRAIN_DATA, size=comp
    ounding(4.0, 32.0, 1.001))
35.         for batch in batches:
36.             texts, annotations = zip(*batch)
```

```
37.             nlp.update(texts, annotations, sgd=op
    timizer, losses=losses)
38.         print("Losses", losses)
39.
40.     # test the trained model
41.     test_text = "Bandeng presto adalah makanan kh
    as Indonesia yang berasal dari Kota Semarang, Jaw
    a Tengah."
42.     doc = nlp(test_text)
43.     print("Tags", [(t.text, t.tag_, t.pos_) for t
    in doc])
44.
45.     # save model to output directory
46.     nlp.to_disk("./pos-tag")
47.     print("Saved model to", output_dir)
48.
49.     # test the save model
50.     print("Loading from", output_dir)
51.     nlp2 = spacy.load("./pos-tag")
52.     doc = nlp2(test_text)
53.     print("Tags", [(t.text, t.tag_, t.pos_) for t
    in doc])
54.
55.
56. if __name__ == "__main__":
57.     plac.call(main)
```

Kode Sumber 4.9 Implementasi pembuatan POS *Tagging*

```
1.  from flask import Flask, request, render_template
    , Markup
2.  import spacy
3.  from spacy import displacy
4.  from setting_color import get_entity_options
5.
6.  @app.route('/insert', methods=['POST'])
7.  def insert():
8.      # mengambil input value dari form
9.      des = request.form['deskripsi']
10.
11.     # load model NER
```

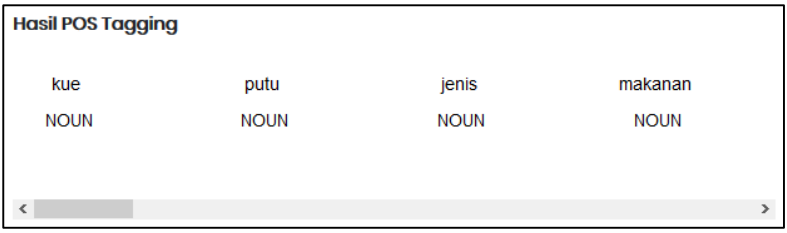
```
12.     nlp = spacy.load(r'C:\Users\ASUS CORNER\model
    _ner_output')
13.
14.     # deteksi NER
15.     doc = nlp(des)
16.
17.     # visualisasi hasil NER
18.     options = get_entity_options()
19.     svg = displacy.render(doc, style='ent', optio
    ns=options)
20.
21.     return render_template('input-
    result.html', svg_ner=Markup(svg))
```

Kode Sumber 4.10 Implementasi NER

4.3.4 Implementasi POS *Tagging*

POS *Tagging* digunakan untuk memberi label pada setiap kata dalam kalimat dengan *tag* yang sesuai dengan kelas kata. Sebelum dilakukan POS *Tagging* perlu dilakukan praproses data untuk menghilangkan tanda baca, *case folding*, dan menghilangkan *stopword* yang dianggap tidak memiliki makna. Maka, pada implementasi POS *Tagging* memanggil fungsi *preprocessing*. Implementasi *preprocessing* dapat dilihat pada Kode Sumber 4.11 dan implementasi POS *Tagging* pada sistem dengan menggunakan Flask dapat dilihat pada Kode Sumber 4.12. Pada kode sumber tersebut, penulis mengambil *value* deskripsi makanan dari *form* deskripsi dengan menggunakan *method request*. Kemudian *load* model POS *Tagging* yang sudah dibuat sebelumnya. Sebelum dilakukan POS *Tagging*, kalimat deskripsi makanan dipraproses terlebih dahulu dengan memanggil fungsi *preprocessing*. Pada fungsi *preprocessing*, kalimat tersebut diubah ke *lower case* dan dihilangkan tanda bacanya dengan menggunakan *regex*. Apabila kalimat tersebut mengandung *double space* maka akan dikonversi menjadi *one space*. Kemudian kalimat yang sudah dibersihkan tersebut ditokenisasi dan dilakukan *stopword removal* untuk

menghilangkan kata-kata yang dianggap tidak memiliki makna. Barulah kemudian kalimat deskripsi makanan yang telah di praproses tersebut masuk ke proses POS *Tagging*. Hasil POS *Tagging* kemudian divisualisasikan dengan menggunakan bantuan pustaka displaCy agar terlihat lebih menarik dan mudah dipahami. Contoh data *input* ditunjukkan oleh Gambar 4.9 dan contoh *output* hasil POS *Tagging* ditunjukkan oleh Gambar 4.11.



Gambar 4.11 Contoh *output* hasil POS *Tagging*

```
1.  import re
2.  from spacy.lang.id import Indonesian
3.
4.  def preprocessing (test_text):
5.      nlp = Indonesian()
6.
7.      # menghilangkan tanda baca dan case folding
8.      remove_punct = re.sub(r"[^a-
    z]", " ", test_text.lower())
9.      # mengubah double space menjadi one space
10.     remove_punct = remove_punct.replace("  ", " "
    )
11.     doc = nlp(remove_punct)
12.
13.     # tokenisasi
14.     token_list = []
15.     for token in doc:
16.         token_list.append(token.text)
17.
18.     # menghilangkan stopword
19.     non_stopword = []
```

```
20.     for word in token_list:
21.         lexeme = nlp.vocab[word]
22.         if lexeme.is_stop == False:
23.             non_stopword.append(word)
24.
25.     # mengubah bentuk tokenisasi menjadi kalimat
    utuh kembali
26.     result = ' '.join(non_stopword)
27.     return result
```

Kode Sumber 4.11 Implementasi *preprocessing*

```
1.  from flask import Flask, request, render_template
    , Markup
2.  from ner_postag_matcher import stopword_removal
3.  import spacy
4.  from spacy import displacy
5.  from setting_color import get_entity_options
6.
7.  @app.route('/insert', methods=['POST'])
8.  def insert():
9.      # mengambil input value dari form
10.     des = request.form['deskripsi']
11.
12.     # load model POS Tagging
13.     nlp2 = spacy.load(r'C:\Users\ASUS CORNER\trai
    n_tagger_v2\pos-tag')
14.
15.     # memanggil fungsi preprocessing untuk
    mendapatkan string input hasil
    praproses dan deteksi tag
16.     doc2 = nlp2(preprocessing (des))
17.
18.     # visualisasi hasil POS Tagging
19.     options = get_entity_options()
20.     svg2 = displacy.render(doc2, style='dep')
21.
22.     return render_template('input-
    result.html', svg_postag=Markup(svg2))
```

Kode Sumber 4.12 Implementasi POS *Tagging*

4.3.5 Implementasi *Rule-based Matching*

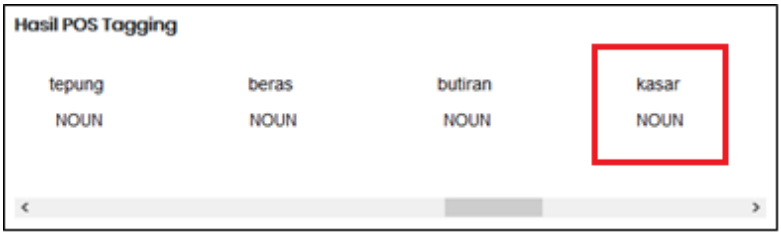
Untuk memperkaya korpus data tentang makanan, selain menggunakan NER, penulis juga menggunakan *Rule-based Matching*. Implementasi *Rule-based Matching* dengan menggunakan model POS *Tagging* (subbab 4.3.2) dapat dilihat pada Kode Sumber 4.13. Hampir sama seperti implementasi POS *Tagging*, implementasi *Rule-based Matching* juga memanggil fungsi *preprocessing*. Pada kode sumber tersebut, awalnya *load* model POS *Tagging* yang sudah dibuat sebelumnya. Kemudian dibuat *rule* untuk mendeteksi entitas bahan. Kalimat deskripsi makanan yang sudah dipraproses dan di-POS *tagging* kemudian dicocokkan dengan *rule* yang sudah dibuat. Hasil yang sesuai akan diambil dan ditampilkan. Contoh data *input* ditunjukkan oleh Gambar 4.9 dan contoh *output* hasil *Rule-based Matching* ditunjukkan oleh Gambar 4.12.

Namun, ternyata hasil dari *Rule-based matching* kurang sesuai sehingga tidak dapat langsung digunakan. Hal tersebut dikarenakan hasil dari POS *Tagging* masih terdapat beberapa kata yang label kelas katanya tidak sesuai seperti ditunjukkan pada Gambar 4.13 dan tidak semua kata yang sesuai dengan *rule* merupakan entitas bahan seperti ditunjukkan pada Gambar 4.14. Oleh karena itu, hasil R*ule-based matching* masih perlu divalidasi secara manual terlebih dahulu untuk mengambil entitas yang benar-benar sesuai.

Hasil Rule-based Matching

['kue', 'kue putu', 'putu', 'putu jenis', 'jenis', 'jenis makanan', 'makanan', 'kue', 'kue isian', 'isian', 'isian gula', 'gula', 'gula jawa', 'jawa', 'parutan', 'parutan kelapa', 'kelapa', 'kelapa tepung', 'tepung', 'tepung beras', 'beras', 'beras butiran', 'butiran', 'butiran kasar', 'kasar', 'kasar kue', 'kue', 'tabung', 'tabung bambu', 'bambu']

Gambar 4.12 Contoh *output* hasil *Rule-based Matching*



Gambar 4.13 Contoh label kelas kata yang tidak sesuai

Gambar 4.14 Contoh kata yang sesuai dengan *rule*

```
import spacy
from spacy.matcher import Matcher

def token_matcher(test_text):
    # load model POS Tagging
    nlp = spacy.load(r"C:\Users\ASUS CORNER\train_tagger_v2\pos-tag")
    matcher = Matcher(nlp.vocab)

    # implementasi rule yang digunakan untuk mendeteksi bahan
    pattern = [{"POS": "NOUN", "OP": "?"}, {"POS": "NOUN", "OP": "?"}]
    matcher.add("BAHAN", None, pattern)

    # memanggil fungsi preprocessing untuk mendapatkan string input hasil praproses
    doc = nlp(preprocessing(test_text))

```

```
16.     # mengambil teks yang sesuai dengan rule
17.     token_match = []
18.     matches = matcher(doc)
19.     for match_id, start, end in matches:
20.         string_id = nlp.vocab.strings[match_id]
    # Get string representation
21.         span = doc[start:end]  # The matched span
22.         if not span.text:
23.             continue
24.         token_match.append(span.text)
25.     return token_match
```

Kode Sumber 4.13 Implementasi *Rule-based Matching*

4.3.6 Implementasi Pembuatan *Dataset* pada Apache Jena Fuseki

Pembuatan *dataset* pada Apache Jena Fuseki membutuhkan bantuan *tools* lain, yaitu Protégé. Protégé merupakan editor ontologi yang *open-source* [17]. Protégé membantu penulis dalam membuat data awal berupa *file* RDF/XML yang nantinya akan diunggah ke Apache Jena Fuseki.

Pada Protégé, penulis membuat ontologi baru yang disimpan dalam tautan http://food-ontology/3#. Dalam ontologi tersebut, penulis membuat empat *annotation property* baru, yaitu: bahan, rasa, cara, dan deskripsi. *Annotation property* ini digunakan untuk menyimpan hasil anotasi dari proses NER. Selain itu, penulis juga membuat *class* baru yang diberi nama ‘makanan’. Setiap data makanan nantinya akan dimasukkan sebagai *instance* baru dalam *class* tersebut. Ontologi yang dibuat kemudian disimpan ke dalam *file* berformat RDF/XML yang selanjutnya dapat digunakan dalam pembuatan *dataset* baru pada Apache Jena Fuseki. *File* RDF/XML dari Protégé dapat dilihat pada Lampiran 1.

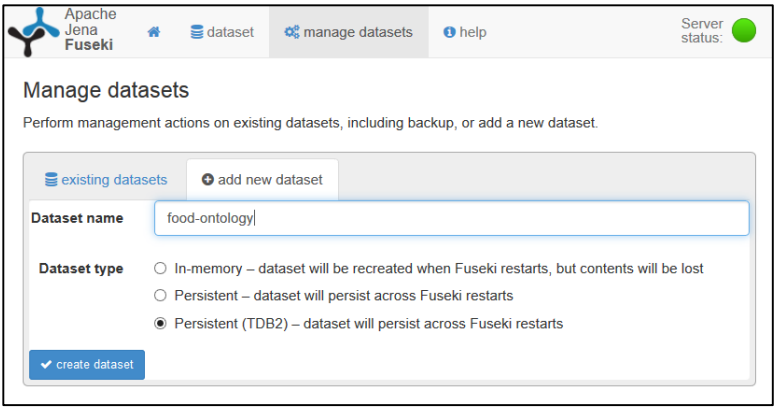
4.3.6.1 Pembuatan *Dataset* Baru pada Apache Jena Fuseki

Cara membuat *dataset* baru pada Apache Jena Fuseki dapat dilihat pada Gambar 4.15. Setelah *dataset* baru berhasil

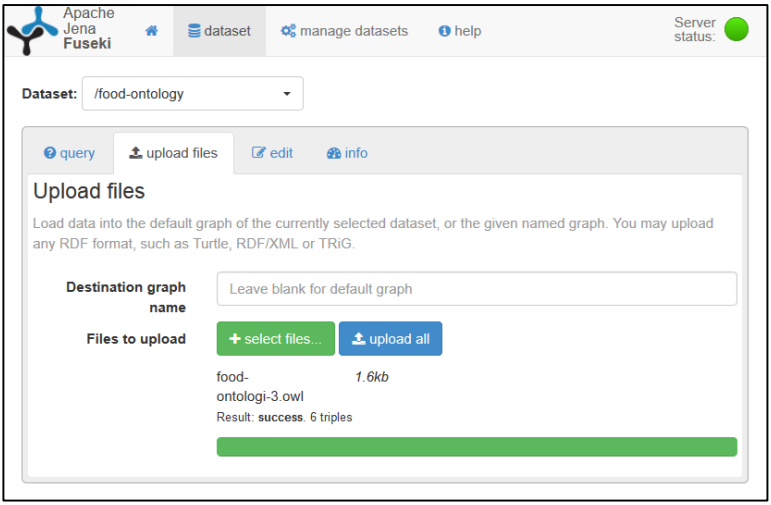
dibuat, selanjutnya mengunggah *file* RDF/XML dari Protégé sebagai data awal seperti ditunjukkan pada Gambar 4.16. Untuk mengecek data yang masuk dapat dijalankan sintaks *query* SPARQL pada *tab query* Apache Jena Fuseki seperti ditunjukkan pada Kode Sumber 4.14. Sintaks *query* SPARQL tersebut bertujuan untuk menampilkan semua pasangan *triplets* (*subject-predicate-object*) yang ada di dalam *dataset*. Hasil *query* dapat dilihat pada Gambar 4.17.

```
PREFIX ont: <http://food-ontology/3#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

SELECT DISTINCT *
WHERE { ?subject ?property ?object }
```

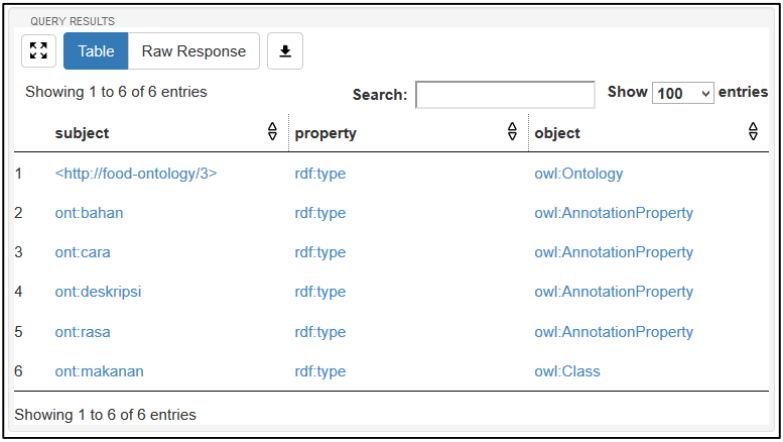
Kode Sumber 4.14 *Query* SPARQL untuk menampilkan semua pasangan *triplets*



Gambar 4.15 Pembuatan *dataset* baru pada Apache Jena Fuseki



Gambar 4.16 Unggah *file* RDF/XML pada Apache Jena Fuseki



Gambar 4.17 Hasil *query* SPARQL pada Apache Jena Fuseki

4.3.6.2 Penyimpanan Anotasi pada Apache Jena Fuseki

Untuk menambah korpus data makanan pada *dataset* Apache Jena Fuseki, penulis menggunakan *script* Python untuk menambah data secara otomatis. Data makanan yang akan diunggah ke Apache Jena Fuseki berupa nama makanan dan deskripsi makanan sebanyak 330 data makanan yang disimpan dalam format .csv. Cara memasukkan data nama makanan dan deskripsi makanan ke Apache Jena Fuseki dapat dilihat pada Kode Sumber 4.15. Sedangkan, cara memasukkan data hasil NER ke Apache Jena Fuseki dapat dilihat pada Kode Sumber 4.16. Data hasil NER dikonversi dulu ke dalam format sintaks SPARQL. Sintaks SPARQL tersebut kemudian digunakan sebagai *query insert* data pada Apache Jena Fuseki. Total data yang berhasil tersimpan pada Apache Jena Fuseki sebanyak 3.910 pasang *triplets*. Contoh pasangan *triplets* hasil anotasi yang tersimpan dalam Apache Jena Fuseki ditunjukkan oleh Gambar 4.18.

Pada Kode Sumber 4.15 dilakukan pembacaan data terhadap semua data makanan yang telah diambil penulis dari berbagai *website* dan disimpan ke dalam *file* berformat CSV. *File* tersebut memiliki 3 kolom, yaitu: label, nama, dan deskripsi. Kolom label digunakan untuk menyimpan nama makanan. Sedangkan kolom nama hampir sama dengan kolom label, tetapi nama makanan yang mengandung spasi diganti dengan *underscore*. Hal ini karena data yang tersimpan dalam kolom nama akan digunakan sebagai nama *instance* makanan. Sementara untuk kolom deskripsi digunakan untuk menyimpan deskripsi makanan. Ketiga kolom tersebut akan diubah menjadi bentuk *list* terlebih dahulu untuk memudahkan proses *insert* data. *List* ini kemudian dikonversi menjadi sintaks *query insert* data. *Query* ini digunakan untuk memasukkan data nama dan deskripsi makanan pada *triple store* Apache Jena Fuseki. Sintaks *query* menggunakan SPARQL tersebut selanjutnya dikirim ke *endpoint* Apache Jena Fuseki

secara berulang sehingga semua data akhirnya dapat tersimpan ke *triple store* Apache Jena Fuseki.

Sementara pada Kode Sumber 4.16 sebenarnya hampir sama dengan Kode Sumber 4.15. Awalnya sama-sama dilakukan pembacaan data makanan yang tersimpan dalam *file* berformat CSV. *File* tersebut memiliki 2 kolom, yaitu nama dan deskripsi. Kedua kolom tersebut diubah menjadi bentuk *list* terlebih dahulu untuk memudahkan proses *insert* data. Selain itu juga dilakukan *load* model NER yang telah dibuat sebelumnya. *List* deskripsi makanan kemudian masuk ke proses deteksi NER menggunakan model NER yang telah di-*load*. Hasil deteksi NER selanjutnya dikonversi menjadi sintaks *query insert* data. *Query* ini digunakan untuk memasukkan entitas bahan, rasa, dan cara memasak makanan pada *triple store* Apache Jena Fuseki. Sintaks *query* menggunakan SPARQL tersebut selanjutnya dikirim ke *endpoint* Apache Jena Fuseki secara berulang sehingga semua hasil NER akhirnya dapat tersimpan ke *triple store* Apache Jena Fuseki.

```
1.  import pandas as pd
2.  from SPARQLWrapper import SPARQLWrapper, JSON
3.
4.  # membaca file csv
5.  df = pd.read_csv("D:\\TC\\TA\\food-
    annotation\\data-all.csv",encoding='cp1252')
6.
7.  # mengubah dataframe menjadi list
8.  des = df['deskripsi'].tolist()
9.  lbl = df['label'].tolist()
10. nm = df['nama'].tolist()
11.
12. for x in range(len(lbl)):
13.     lbl[x] = '"'"'"+str(lbl[x])+'"'
14.     des[x] = '"'"'"+str(des[x])+'"'
15.
16. for x in range(len(nm)):
```

```
17.     # insert data nama makanan
    dan deskripsi makanan ke jena fuseki
18.     sparql = SPARQLWrapper("http://localhost:3030
    /food-ontology-3-1/update")
19.     sparql.setQuery("""
20.                 PREFIX ont: <http://food-
    ontology/3#>
21.                 PREFIX rdf: <http://www.w3.org/19
    99/02/22-rdf-syntax-ns#>
22.                 PREFIX owl: <http://www.w3.org/20
    02/07/owl#>
23.                 PREFIX xsd: <http://www.w3.org/20
    01/XMLSchema#>
24.                 PREFIX rdfs: <http://www.w3.org/2
    000/01/rdf-schema#>
25.
26.                 INSERT DATA { ont:"""+nm[x]+""" r
    df:type ont:makanan };
27.                 INSERT DATA { ont:"""+nm[x]+""" r
    dfs:label """+lbl[x]+""" };
28.                 INSERT DATA { ont:"""+nm[x]+""" o
    nt:deskripsi """+des[x]+""" };
29.             """)
30.
31.     sparql.setReturnFormat(JSON)
32.     results = sparql.query().convert()
33.     print(results)
```

Kode Sumber 4.15 Memasukkan data nama dan deskripsi makanan ke Apache Jena Fuseki

```
1.  import pandas as pd
2.  import spacy
3.  from SPARQLWrapper import SPARQLWrapper, JSON
4.
5.  # load model NER
6.  nlp = spacy.load(r'C:\Users\ASUS CORNER\model_ner
    _output')
7.
8.  # membaca file csv
9.  df = pd.read_csv("D:\\TC\\TA\\food-
    annotation\\data-all.csv",encoding='cp1252')
```

```
10.
11. # mengubah dataframe ke dalam list
12. des = df['deskripsi'].tolist()
13. nm = df['nama'].tolist()
14.
15. # menyimpan hasil anotasi NER
16. result=[]
17. for text in des:
18.     doc = nlp(text)
19.     result.append([(ent.text.lower(), ent.label_.
    lower()) for ent in doc.ents if ent.label_])
20.
21. for x in range(len(result)):
22.     ent_text = []
23.     ent_label = []
24.     for i in range(len(result[x])):
25.         if "nama" in result[x][i]:
26.             continue
27.         ent_text.append('"'+result[x][i][0]+
    '"')
28.         ent_label.append("ont:"+result[x][i][1])

29.     insert = ""
30.     for i in range(len(ent_text)):
31.         text = ("INSERT DATA { ont:"+nm[x]+" "+en
    t_label[i]+" "+ent_text[i]+"};")
32.         insert += text
33.
34.     # insert data hasil anotasi NER ke jena fuse-
    ki
35.     sparql = SPARQLWrapper("http://localhost:3030
    /food-ontology-3/update")
36.     sparql.setQuery("""
37.                 PREFIX ont: <http://food-
    ontology/3#>
38.                 PREFIX rdf: <http://www.w3.org/19
    99/02/22-rdf-syntax-ns#>
39.                 PREFIX owl: <http://www.w3.org/20
    02/07/owl#>
40.                 PREFIX xsd: <http://www.w3.org/20
    01/XMLSchema#>
```

```
41.                 PREFIX rdfs: <http://www.w3.org/2
   000/01/rdf-schema#>
42.
43.                 """+insert+"""
44.             """)
45.
46.     sparql.setReturnFormat(JSON)
47.     results = sparql.query().convert()
48.     print(results)
```

Kode Sumber 4.16 Memasukkan data hasil anotasi NER ke Apache Jena Fuseki

	subject	property	object
1	<http://food-ontology/3>	rdf:type	owl:Ontology
2	ont:bahan	rdf:type	owl:AnnotationProperty
3	ont:cara	rdf:type	owl:AnnotationProperty
4	ont:deskripsi	rdf:type	owl:AnnotationProperty
5	ont:rasa	rdf:type	owl:AnnotationProperty
6	ont:makanan	rdf:type	owl:Class
7	ont:bandeng_presto	rdf:type	ont:makanan
8	ont:bandeng_presto	ont:bahan	"ikan bandeng"
9	ont:bandeng_presto	ont:bahan	"bawang putih"
10	ont:bandeng_presto	ont:bahan	"kunyit"
11	ont:bandeng_presto	ont:bahan	"garam"
12	ont:bandeng_presto	ont:cara	"presto"
13	ont:bandeng_presto	ont:deskripsi	"Bandeng presto adalah makanan khas Indonesia yang berasal dari Kota Semarang, Jawa Tengah. Makanan ini dibuat dari ikan bandeng yang dibumbui dengan bawang putih, kunyit dan garam. Ikan bandeng ini kemudian dimasak pada alas daun pisang dengan cara presto."
14	ont:bandeng_presto	rdfs:label	"bandeng presto"

Gambar 4.18 Contoh *triplets* pada Apache Jena Fuseki

4.3.7 Implementasi *Insert* Data ke Apache Jena Fuseki

Nama makanan dan hasil anotasi NER deskripsi makanan digunakan sebagai masukan untuk *insert* data ke dalam *triple store* Apache Jena Fuseki. Nama makanan akan dijadikan sebagai nama *instance*. Karena nama *instance* tidak boleh mengandung spasi, maka untuk nama makanan yang mengandung spasi akan otomatis diganti dengan *underscore*. *Instance* makanan tersebut memiliki label nama makanan dan *annotation property* deskripsi yang berisi deskripsi makanan. Untuk hasil anotasi NER deskripsi makanan akan dijadikan *annotation property* sesuai dengan labelnya. Implementasi *insert* data ke dalam Apache Jena Fuseki dapat dilihat pada Kode Sumber 4.17. Pada kode sumber tersebut, awalnya *load* model NER yang sudah dibuat. Kemudian dilakukan deteksi NER pada kalimat deskripsi makanan menggunakan model NER yang sudah dibuat. Hasil deteksi NER selanjutnya dikonversi menjadi sintaks *query insert* data. Sintaks *insert* data menggunakan SPARQL tersebut kemudian dikirim ke *endpoint* Apache Jena Fuseki sehingga hasil NER dapat tersimpan ke *triple store* Apache Jena Fuseki. Contoh data *input* ditunjukkan oleh Gambar 4.9 dan contoh *output* hasil *insert* data ke Apache Jena Fuseki ditunjukkan oleh Gambar 4.19.

```
1.  import spacy
2.  from SPARQLWrapper import SPARQLWrapper, JSON
3.
4.  def insert_fuseki(nama, label, deskripsi):
5.      label = '"'"'+label+'"'
6.      deskripsi = '"'"'+deskripsi+'"'
7.
8.      # load model NER
9.      nlp = spacy.load(r'C:\Users\ASUS CORNER\model
    _ner_output')
10.     doc = nlp(deskripsi)
11.     result = []
12.     # hasil NER
```

```
13.     result.append([(ent.text.lower(), ent.label_.
    lower()) for ent in doc.ents if ent.label_])
14.
15.     ent_text = []
16.     ent_label = []
17.     for i in range(len(result[0])):
18.         if "nama" in result[0][i]:
19.             continue
20.         ent_text.append('"'+result[0][i][0]+
    '"')
21.         ent_label.append("ont:"+result[0][i][1])
22.
23.     insert = ""
24.     for i in range(len(ent_text)):
25.         text = ("INSERT DATA { ont:"+nama+" "+ent
    _label[i]+" "+ent_text[i]+"};")
26.         insert += text
27.
28.     # insert data ke jena fuseki
29.     sparql = SPARQLWrapper("http://localhost:3030
    /food-ontology-3/update")
30.     sparql.setQuery("""
31.             PREFIX ont: <http://food-
    ontology/3#>
32.             PREFIX rdf: <http://www.w3.org/1999/0
    2/22-rdf-syntax-ns#>
33.             PREFIX owl: <http://www.w3.org/2002/0
    7/owl#>
34.             PREFIX xsd: <http://www.w3.org/2001/X
    MLSchema#>
35.             PREFIX rdfs: <http://www.w3.org/2000/
    01/rdf-schema#>
36.
37.             INSERT DATA { ont:"""+nama+""" rdf:ty
    pe ont:makanan };
38.             INSERT DATA { ont:"""+nama+""" rdfs:l
    abel """+label+""" };
39.             INSERT DATA { ont:"""+nama+""" ont:de
    skripsi """+deskripsi+""" };
40.             """+insert+"""
41.         """)
42.
```

```
43.     sparql.setReturnFormat(JSON)
44.     results = sparql.query().convert()
45.     return(results)
```

Kode Sumber 4.17 Implementasi *insert* data ke Apache Jena Fuseki

	subject	property	object
1	ont:kue_putu	rdf:type	ont:makanan
2	ont:kue_putu	ont:bahan	"gula jawa"
3	ont:kue_putu	ont:bahan	"tepung beras"
4	ont:kue_putu	ont:bahan	"parutan kelapa"
5	ont:kue_putu	ont:cara	"dikukus"
6	ont:kue_putu	ont:deskripsi	"Kue putu adalah jenis makanan tradisional nusantara yang berupa kue dengan isian gula jawa, dibalut dengan parutan kelapa, dan tepung beras butiran kasar. Kue ini dikukus dengan diletakkan di dalam tabung bambu yang sedikit dipadatkan."
7	ont:kue_putu	rdfs:label	"kue putu"

Gambar 4.19 Contoh hasil *insert* data ke Apache Jena Fuseki

4.3.8 Implementasi *Query* dari Apache Jena Fuseki

Pengguna dapat melakukan *query* atau pencarian berdasarkan nama, bahan, rasa, dan cara memasak makanan. Kata kunci pencarian yang di-*input* pengguna akan di-*split* dan kemudian dilakukan *query* ke Apache Jena Fuseki secara berulang untuk mengecek setiap kata kunci pencarian. Implementasi *query* dari Apache Jena Fuseki dapat dilihat pada Kode Sumber 4.18. Pada kode sumber tersebut, kata kunci pencarian atau *keyword* akan di-*split* (dipisahkan tiap katanya menjadi sebuah *string*). Setiap *string keyword* tersebut kemudian dicocokkan dengan data yang tersimpan di *triple store* Apache Jena Fuseki. Cara

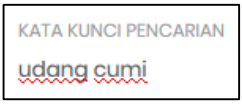
mencocokkannya adalah dengan melakukan *query* secara berulang untuk mengecek nama, bahan, rasa, dan cara memasak makanan yang sesuai dengan kata kunci pencarian. *Query* pada Apache Jena Fuseki diimplementasikan dengan menggunakan *statement select* menggunakan SPARQL. Hasil pencarian yang didapatkan dari Apache Jena Fuseki memiliki format JSON sehingga *string* dan tanda baca yang tidak penting harus dihilangkan terlebih dahulu agar tersisa nama makanannya saja. Kemudian nama makanan yang benar-benar cocok dengan semua kata kunci akan ditampilkan. Contoh kata kunci pencarian ditunjukkan oleh Gambar 4.20 dan contoh *output* hasil *query* dari Apache Jena Fuseki ditunjukkan oleh Gambar 4.21.

```
1.  import requests
2.
3.  def query_fuseki(keyword):
4.      # split kata kunci pencarian setiap ada spasi

5.      key = keyword.split()
6.      result_list = []
7.      # query ke jena fuseki secara berulang untuk
    mengecek setiap kata kunci pencarian
8.      for i in range(len(key)):
9.          str_query = 'PREFIX ont: <http://food-
    ontology/3#> \
10.                     PREFIX rdf: <http://www.w3.or
    g/1999/02/22-rdf-syntax-ns#> \
11.                     PREFIX owl: <http://www.w3.or
    g/2002/07/owl#> \
12.                     PREFIX xsd: <http://www.w3.or
    g/2001/XMLSchema#> \
13.                     PREFIX rdfs: <http://www.w3.o
    rg/2000/01/rdf-schema#> \
14.                     \
15.                     SELECT DISTINCT ?nama \
16.                     WHERE { \
17.                         { { ?subject rdfs:label ?
    object. \
```

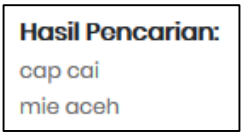
```
18.                         FILTER regex(?object, "'+key[i]+'") } \
19.                         UNION \
20.                         { ?subject ont:bahan ?object. \
21.                         FILTER regex(?object, "'+key[i]+'") } \
22.                         UNION \
23.                         { ?subject ont:rasa ?object. \
24.                         FILTER regex(?object, "'+key[i]+'") } \
25.                         UNION \
26.                         { ?subject ont:cara ?object. \
27.                         FILTER regex(?object, "'+key[i]+'") } } .\
28.                         ?subject rdfs:label ?nama \
29.                     }'
30.
31.         response = requests.post('http://localhost:3030/food-ontology-3/query', data={'query':str_query})
32.
33.         # menyimpan nama makanan yang sesuai dengan kata kunci pencarian
34.         resp_json = str(response.json())
35.         word = "value"
36.         if word in resp_json:
37.             resp_json = resp_json.replace("'}}, {'nama': {'type': 'literal', 'value': '", ", ")
38.             res = resp_json.replace("{'head': {'vars': ['nama']}, 'results': {'bindings': [{'nama': {'type': 'literal', 'value': '", "")
39.             result = res.replace("'}}]}}", "\n")

40.         else:
41.             result = "Tidak ditemukan"
42.         result_list.append(result)
43.
```

```
44.     # menyimpan semua nama makanan dari hasil pen
    carian semua kata kunci
45.     all_list = []
46.     for i in range(len(key)):
47.         result_list[i] = result_list[i].split(",
    ")
48.         all_list.append(result_list[i])
49.
50.     # mengambil nama makanan yang benar-
    benar cocok dengan semua kata kunci pencarian
51.     common = set(all_list[0])
52.     for s in all_list[1:]:
53.         common.intersection_update(s)
54.     str_result = ", ".join(common)
55.     return (str_result)
```

Kode Sumber 4.18 Implementasi *query* dari Apache Jena Fuseki



Gambar 4.20 Contoh kata kunci pencarian



Gambar 4.21 Contoh *output* hasil *query* dari Apache Jena Fuseki

4.3.9 Implementasi Unduh *File* Hasil Anotasi dari Apache Jena Fuseki

Hasil anotasi yang disimpan di Apache Jena Fuseki dapat diunduh dalam format JSON. Cara menyimpan hasil anotasi dari Apache Jena Fuseki ke dalam file JSON dapat dilihat pada Kode Sumber 4.19. Pada kode sumber tersebut, awalnya penulis mengambil data hasil anotasi dari Apache Jena Fuseki dengan

query select menggunakan SPARQL. Hasil anotasi kemudian dikonversi menjadi format JSON dan ditulis ke *file* JSON. Implementasi unduh *file* JSON ditunjukkan pada Kode Sumber 4.20. Pada kode sumber tersebut, didapatkan *path* lokasi *file* JSON hasil anotasi yang kemudian dikirim sebagai *attachment* sehingga nantinya pengguna dapat mengunduh *file* tersebut. Contoh data *input* ditunjukkan oleh Gambar 4.9 dan contoh *file* JSON *output* hasil anotasi dari Apache Jena Fuseki dapat dilihat pada Lampiran 3.

```
1.  import requests,json
2.
3.  def relasi_fuseki(text):
4.      # mengambil data hasil anotasi dari jena fuseki
5.      str_query = 'PREFIX ont: <http://food-ontology/3#> \
6.                  PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> \
7.                  PREFIX owl: <http://www.w3.org/2002/07/owl#> \
8.                  PREFIX xsd: <http://www.w3.org/2001/XMLSchema#> \
9.                  PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#> \
10.                 \
11.                 SELECT DISTINCT * \
12.                 WHERE { \
13.                     { ?subject rdfs:label "'+text+'". \
14.                       ?subject ?property ?object } \
15.                     UNION \
16.                     { ?subject rdfs:label "'+text+'". \
17.                       ?subject ?property ?object. \
18.                       ?object ?property2 ?type } \
19.                     UNION \
```

```
20.                     { ?subject rdfs:label "'+text
   +'". \
21.                       ?subject ?property ?object.
   \
22.                       ?property ?property2 ?type
   } \
23.                 }'
24.
25.     response = requests.post('http://localhost:30
   30/food-ontology-
   3/query', data={'query':str_query})
26.     resp = response.json()
27.
28.     # konversi ke format json
29.     result = json.dumps(resp)
30.     # hasil ditulis ke file json
31.     text_file = open("output.json", "w")
32.     text_file.write(result)
33.     text_file.close()
```

Kode Sumber 4.19 Menyimpan hasil anotasi dari Apache Jena Fuseki ke dalam *file* JSON

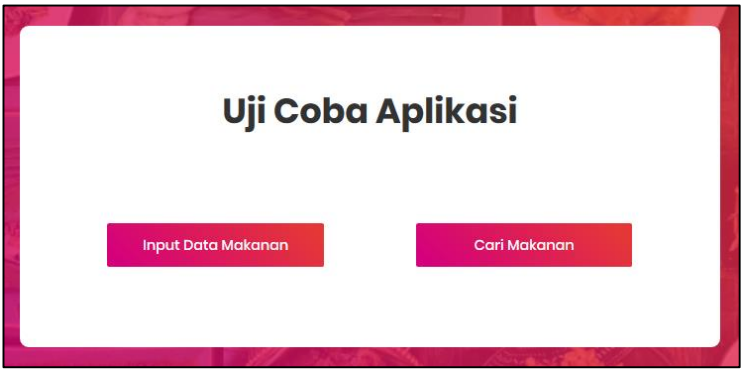
```
1.  import os
2.  from flask import send_file
3.
4.  def download():
5.      path = os.getcwd()
6.      filepath = os.path.join(path,'output.json')
7.      return send_file(filepath,as_attachment=True)
```

Kode Sumber 4.20 Implementasi unduh *file* JSON

4.4 Implementasi Antarmuka Sistem

Implementasi antarmuka sistem dibuat untuk mempermudah uji coba dan evaluasi. Implementasi antarmuka sistem dibuat dengan menggunakan kerangka kerja Flask dan *triple store* Apache Jena Fuseki. Pada halaman depan atau beranda aplikasi seperti ditunjukkan pada Gambar 4.22, terdapat dua tombol yang memiliki

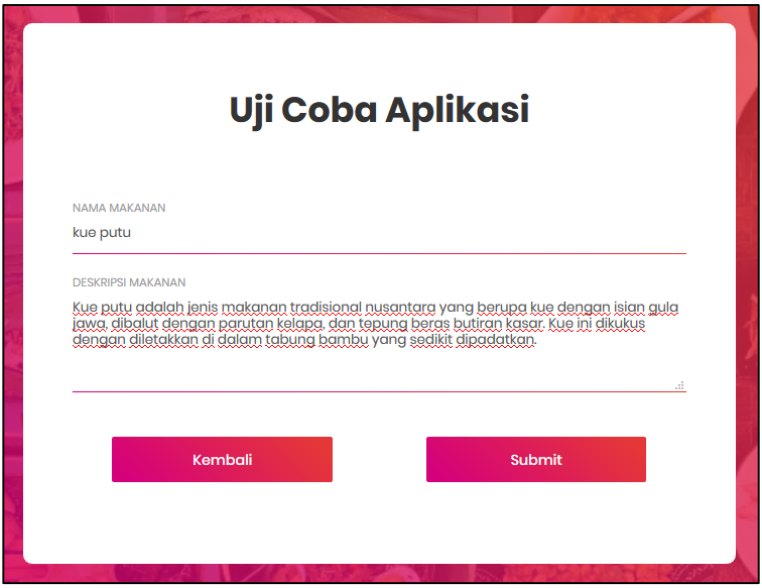
fungsi berbeda, yang pertama untuk *input* data makanan dan yang kedua untuk mencari makanan.



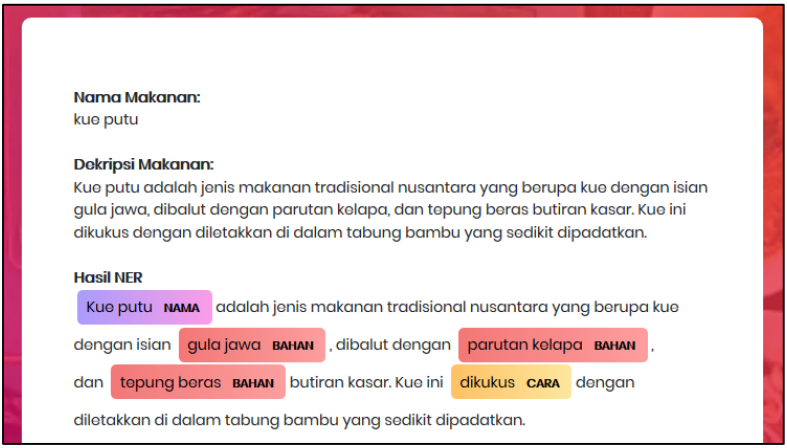
Gambar 4.22 Halaman beranda aplikasi

Pada halaman *input* data makanan, pengguna dapat memasukkan nama dan deskripsi makanan pada *form* seperti ditunjukkan pada Gambar 4.23. Setelah memasukkan nama dan deskripsi makanan, pengguna harus menekan tombol Submit agar sistem memproses NER, POS *Tagging*, dan *Rule-based Matching* serta menyimpan hasil anotasi NER pada *triple store* Apache Jena Fuseki. Hasil NER, POS *Tagging*, dan *Rule-based Matching* dapat dilihat pada Gambar 4.24 dan Gambar 4.25. Pada bagian kanan bawah halaman tersebut terdapat tombol unduh *file* JSON yang jika diklik dapat memunculkan *pop up* seperti ditunjukkan pada Gambar 4.26. *File* JSON tersebut berisi hasil anotasi dari *triple store* Apache Jena Fuseki.

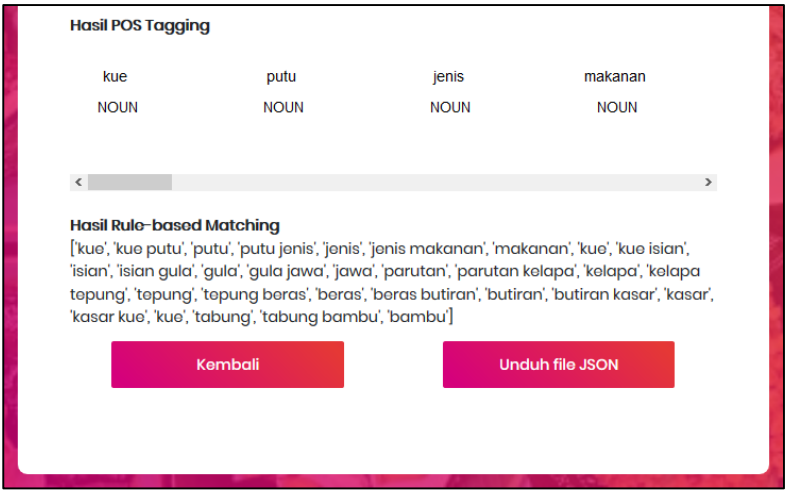
Sedangkan pada halaman cari makanan, pengguna dapat memasukkan kata kunci pencarian pada *form* seperti ditunjukkan pada Gambar 4.27. Kata kunci pencarian dapat berupa nama, bahan, rasa, ataupun cara memasak makanan. Bisa juga berupa gabungan dua atau lebih dari keempatnya.



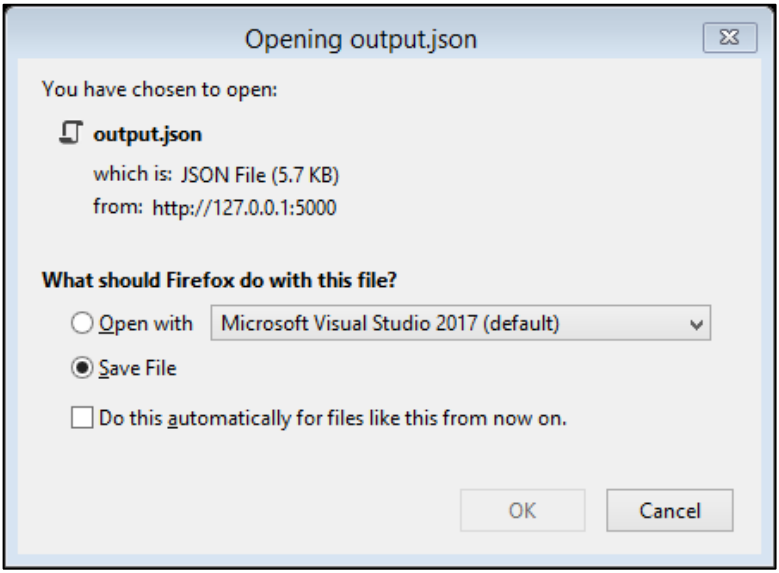
Gambar 4.23 Halaman *input* data makanan



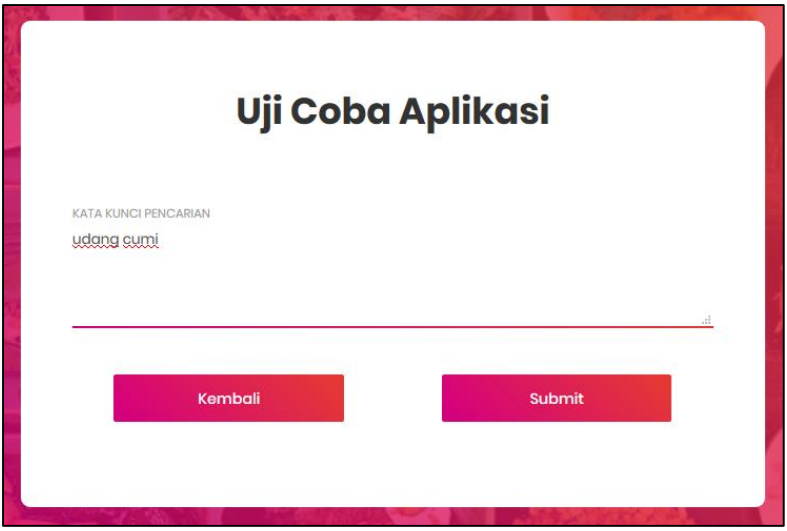
Gambar 4.24 Halaman hasil NER



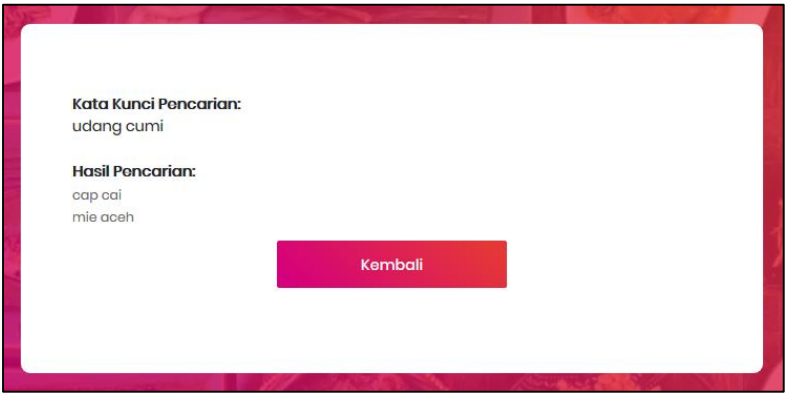
Gambar 4.25 Halaman hasil POS *Tagging* dan *Rule-based Matching*



Gambar 4.26 *Pop up* unduh *file* JSON



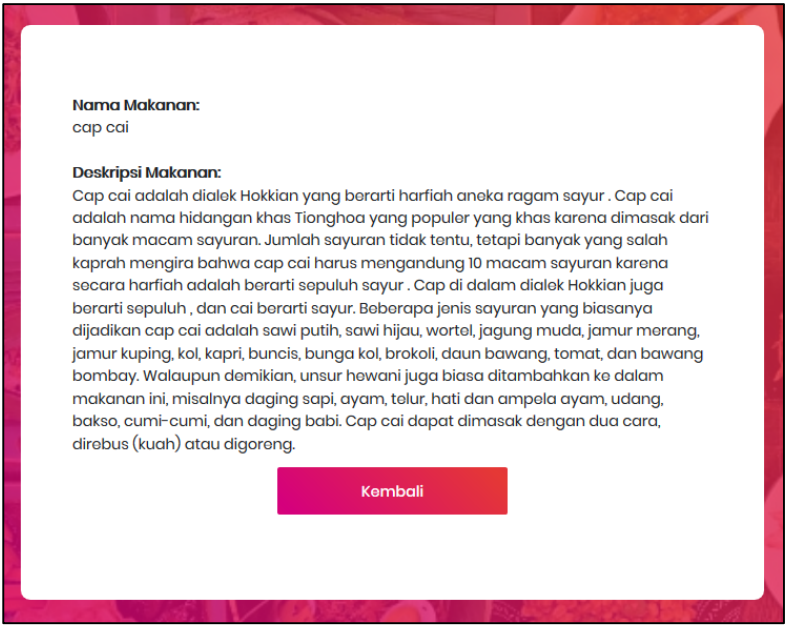
Gambar 4.27 Halaman cari makanan



Gambar 4.28 Halaman hasil pencarian makanan

Setelah memasukkan kata kunci sesuai dengan keinginan, pengguna harus menekan tombol Submit agar sistem memproses

pencarian. Halaman hasil pencarian makanan yang sesuai dengan kata kunci dapat dilihat pada Gambar 4.28. Hasil pencarian berupa nama makanan yang jika diklik dapat merujuk ke halaman deskripsi makanan seperti ditunjukkan pada Gambar 4.29.



Gambar 4.29 Halaman deskripsi makanan

BAB V
UJI COBA DAN EVALUASI

Pada bab ini, akan dijabarkan hasil uji coba beserta evaluasi dari sistem yang telah dibuat. Pengujian dilakukan untuk mengetahui kinerja sistem dengan lingkungan uji coba yang telah ditentukan.

5.1. Lingkungan Uji Coba

Lingkungan uji coba pada pengerjaan tugas akhir ini dilakukan pada perangkat keras dan sistem operasi sebagai berikut:

- Processor Intel® Core™ i7-4720HQ CPU @ 2.60GHz
- Installed RAM 12.0 GB (11.9 GB usable)
- System Type 64-bit operating system, x64-based processor
- Windows Edition Windows 8.1 Pro

5.2. Data Uji Coba

Data uji coba yang digunakan untuk evaluasi model NER merupakan data masukan baru (di luar data latih dan *dataset* yang sudah ada) dengan total sejumlah 232 kalimat deskripsi makanan. Untuk evaluasi *input* data dan bentuk anotasi juga menggunakan data masukan baru berupa nama dan deskripsi makanan. Sedangkan untuk evaluasi *query* label penulis menggunakan beberapa kata kunci pencarian yang berbeda.

5.3. Skenario Uji Coba

Subbab ini akan menjelaskan skenario uji yang telah dilakukan. Terdapat 5 skenario uji coba yang dilakukan. Berikut merupakan penjelasan setiap skenario pengujian:

1. Skenario Pengujian 1

   Dalam skenario ini akan dilakukan pengujian terhadap model NER secara manual. Evaluasi model NER dilakukan dengan menghitung nilai *accuracy, precision, recall,* dan *f-measure* untuk setiap label NER.

2. Skenario Pengujian 2
   Dalam skenario ini akan dilakukan pengujian dengan data masukan baru berupa nama dan deskripsi makanan ke dalam sistem.
3. Skenario Pengujian 3
   Dalam skenario ini akan dilakukan pengujian dengan melakukan *query* label. Pengujian dilakukan dengan memasukkan kata kunci yang berupa nama, bahan, rasa, atau cara memasak makanan.
4. Skenario Pengujian 4
   Dalam skenario ini akan dilakukan pengujian bentuk anotasi dari SPARQL Apache Jena Fuseki yang disimpan dalam *file* berformat JSON. Pengujian ini dilakukan untuk memastikan apakah bentuk anotasi sudah sesuai dengan standar atau belum.
5. Skenario Pengujian 5
   Dalam skenario ini akan dilakukan pengujian sistem oleh *end user*. Pengujian ini dilakukan untuk menilai apakah sistem yang dibuat dapat membantu pengguna dalam mencari kuliner yang diinginkan atau belum.

5.3.1 Skenario Pengujian 1

Pada skenario ini dilakukan evaluasi model NER secara manual. Evaluasi model NER dilakukan dengan menghitung nilai *accuracy*, *precision*, *recall*, dan *f-measure* untuk setiap entitas label NER, yaitu NAMA, BAHAN, RASA, dan CARA. Model NER yang akan dievaluasi adalah model NER yang telah dibuat sebelumnya seperti dijelaskan pada subbab 4.3.1.

Data uji coba yang digunakan untuk skenario pengujian ini merupakan data masukan baru (di luar data latih) berupa deskripsi makanan yang belum teranotasi label entitas NER. Terdapat 3 data uji yang digunakan untuk evaluasi model NER seperti ditunjukkan pada Tabel 5.1.

Tabel 5.1 Data uji evaluasi model NER

Nama	Jumlah Data Uji	Keterangan
Data Uji 1	145 kalimat dari 16 makanan	Nama makanan ada di dalam data latih namun berbeda deskripsi
Data Uji 2	87 kalimat dari 11 makanan	Nama dan deskripsi makanan di luar data latih
Data Uji 3	232 kalimat dari 27 makanan	Data gabungan dari Data Uji 1 dan Data Uji 2

Data uji tersebut kemudian dianotasi otomatis dengan menggunakan model NER yang telah dibuat sebelumnya. Pengujian dilakukan dengan menghitung secara manual nilai *True Positive* (TP), *True Negative* (TN), *False Positive* (FP), dan *False Negative* (FN) dari setiap entitas NER. Hasil evaluasi model NER menggunakan Data Uji 1 dapat dilihat pada Tabel 5.2. Hasil evaluasi model NER menggunakan Data Uji 2 dapat dilihat pada Tabel 5.3. Hasil evaluasi model NER menggunakan Data Uji 3 dapat dilihat pada

Tabel 5.4. Grafik evaluasi model NER untuk semua label menggunakan semua data uji ditunjukkan oleh Gambar 5.1.

Tabel 5.2 Hasil evaluasi model NER menggunakan Data Uji 1

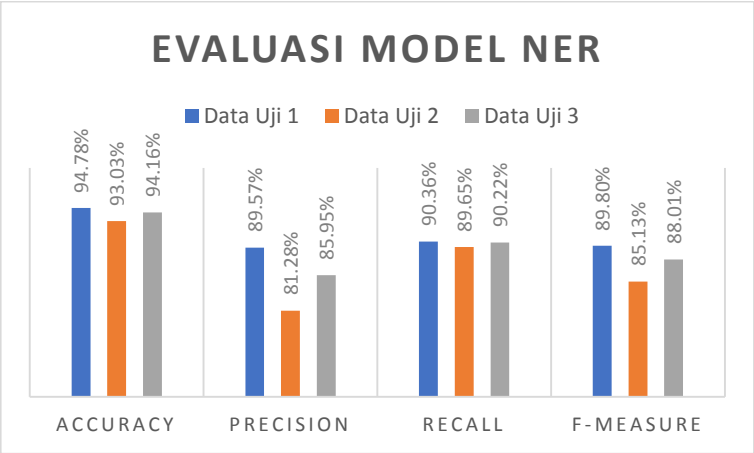
Entitas	Perhitungan secara Manual			
	Accuracy	***Precision***	***Recall***	***F-Measure***
NAMA	91.41%	80.36%	85.71%	82.95%
BAHAN	88.63%	85.07%	96.20%	90.30%
RASA	99.77%	100.00%	92.86%	96.30%
CARA	99.30%	92.86%	86.67%	89.65%

Tabel 5.3 Hasil evaluasi model NER menggunakan Data Uji 2

Entitas	Perhitungan secara Manual			
	Accuracy	***Precision***	***Recall***	***F-Measure***
NAMA	86.67%	75.38%	76.56%	75.97%
BAHAN	87.55%	85.12%	90.35%	87.66%
RASA	99.14%	80.00%	100.00%	88.89%
CARA	98.71%	84.61%	91.67%	88.00%

Tabel 5.4 Hasil evaluasi model NER menggunakan Data Uji 3

Entitas	Perhitungan secara Manual			
	Accuracy	***Precision***	***Recall***	***F-Measure***
NAMA	89.76%	78.53%	82.25%	80.35%
BAHAN	88.25%	85.09%	94.30%	89.46%
RASA	99.55%	91.30%	95.45%	93.33%
CARA	99.10%	88.89%	88.89%	88.89%



Gambar 5.1 Grafik evaluasi model NER untuk semua label menggunakan semua data uji

5.3.2 Skenario Pengujian 2

Pada skenario ini dilakukan uji coba *input* data nama dan deskripsi makanan menggunakan data masukan baru ke dalam sistem. Data masukan baru merupakan data uji coba di luar data latih dan *dataset* yang sudah ada. Kondisi sistem sebelum dilakukan pengujian terdapat 3.910 pasang *triplets* yang tersimpan dalam *triple store* Apache Jena Fuseki seperti yang telah dijelaskan pada subbab 4.3.7.

Data baru yang akan diuji terdapat pada Tabel 5.5. Hasil uji coba *input* data menggunakan Data Uji 4 dapat dilihat pada Gambar 5.2 dan hasil yang tersimpan pada Apache Jena Fuseki ditunjukkan oleh Gambar 5.3. Hasil uji coba *input* data menggunakan Data Uji 5 dapat dilihat pada Gambar 5.4 dan hasil yang tersimpan pada Apache Jena Fuseki ditunjukkan oleh Gambar 5.5. Hasil uji coba *input* data menggunakan Data Uji 6 dapat dilihat pada Gambar 5.6 dan hasil yang tersimpan pada Apache Jena Fuseki ditunjukkan oleh Gambar 5.7 dan Gambar 5.8.

Tabel 5.5 Data uji coba untuk *input* data

Nama	Data
Data Uji 4	Nama makanan: kue dadar gulung Deskripsi makanan: Kue dadar gulung merupakan penganan khas Indonesia yang dapat digolongkan sebagai panekuk yang diisi dengan parutan kelapa yang dicampur dengan gula jawa cair. Isi ini disebut unti. Kulit dadar gulung berwarna hijau karena diberi pewarna daun suji.
Data Uji 5	Nama makanan: krengsengan

Nama	Data
	Deskripsi makanan: Krengsengan adalah masakan yang biasanya terbuat dari daging kambing yang dipotong kecil (2X2 cm), yang diberi bumbu tumisan bawang merah, bawang putih, ketumbar, merica, pala, (dihaluskan terlebih dahulu) ditambah dengan kecap manis pada waktu memasak, sehingga tidak banyak berkuah dan berwarna coklat tua, ada juga yang dicampur dengan bagian jeroan kambing. Masakan yang khas menggunakan daging kambing ini biasanya dijumpai bersama gulai kambing, dan tongseng yang juga menggunakan daging kambing. Masakan ini umumnya dikenal di Jawa Timur, Jawa Tengah dan DI Yogyakarta.
Data Uji 6	Nama makanan: nasi goreng Deskripsi makanan: Nasi goreng adalah sebuah makanan berupa nasi yang digoreng dan diaduk dalam minyak goreng atau margarin. Bahan utama nasi goreng antara lain nasi yang telah masak, kecap manis, bubuk terasi (pasta udang), garam, bawang putih, bawang merah, cabe rawit, daun bawang, pala, kunyit, minyak sayur, bawang merah, gula, pasta jahe, dan irisan mentimun serta tomat untuk hiasan. Beberapa resep bisa menambahkan lada hitam, saus ikan, atau kaldu bubuk sebagai penambah bumbu dan rasa. Biasanya juga ditambahkan telur, ayam, dan kerupuk sebagai pelengkap.

Hasil NER

Kue dadar gulung merupakan penganan khas Indonesia yang dapat digolongkan sebagai panekuk yang diisi dengan parutan kelapa BAHAN yang dicampur dengan gula jawa cair BAHAN . Isi ini disebut unti. Kulit dadar gulung berwarna hijau karena diberi pewarna daun suji BAHAN .

Hasil POS Tagging

kue	dadar	gulung	penganan
NOUN	NOUN	VERB	NOUN

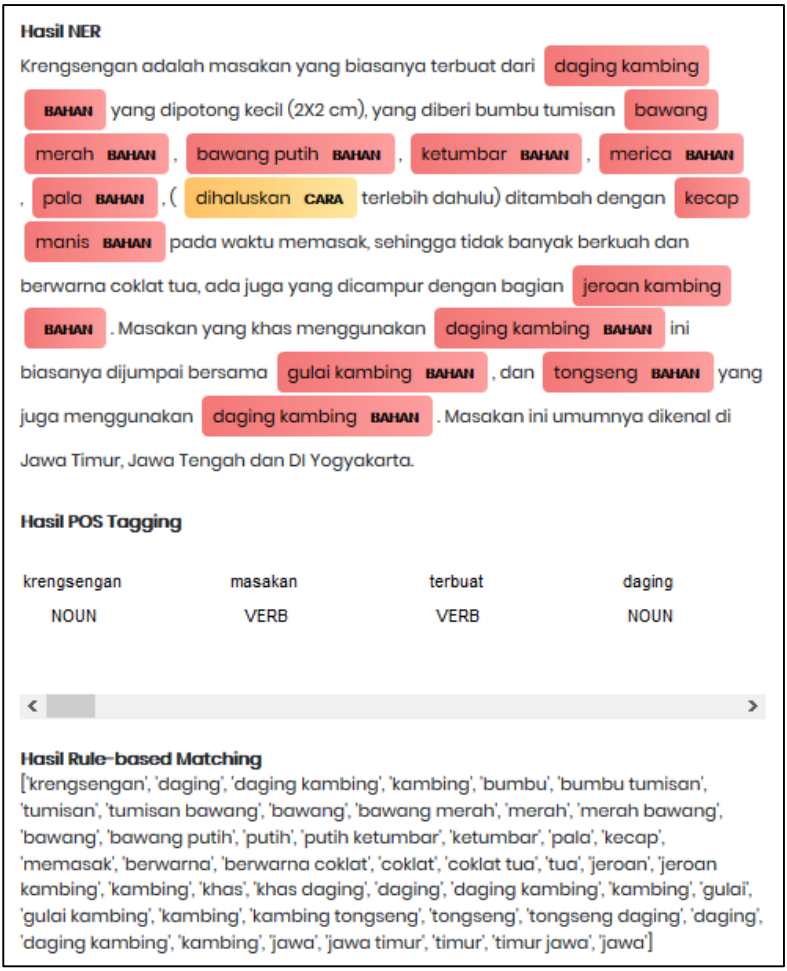
Hasil Rule-based Matching

['kue', 'kue dadar', 'dadar', 'penganan', 'panekuk', 'parutan', 'parutan kelapa', 'kelapa', 'gula', 'gula jawa', 'jawa', 'jawa cair', 'cair', 'cair isi', 'isi', 'isi unti', 'unti', 'unti kulit', 'kulit', 'kulit dadar', 'dadar', 'dadar gulung', 'gulung', 'gulung berwarna', 'berwarna', 'berwarna hijau', 'hijau', 'hijau pewarna', 'pewarna', 'pewarna daun', 'daun', 'daun suji', 'suji']

Gambar 5.2 Hasil uji coba *input* data menggunakan Data Uji 4

	subject	property	object
1	ont:kue_dadar_gulung	rdf:type	ont:makanan
2	ont:kue_dadar_gulung	ont:bahan	"parutan kelapa"
3	ont:kue_dadar_gulung	ont:bahan	"gula jawa cair"
4	ont:kue_dadar_gulung	ont:bahan	"pewarna daun suji"
5	ont:kue_dadar_gulung	ont:deskripsi	"Kue dadar gulung merupakan penganan khas Indonesia yang dapat digolongkan sebagai panekuk yang diisi dengan parutan kelapa yang dicampur dengan gula jawa cair. Isi ini disebut unti. Kulit dadar gulung berwarna hijau karena diberi pewarna daun suji."
6	ont:kue_dadar_gulung	rdfs:label	"kue dadar gulung"

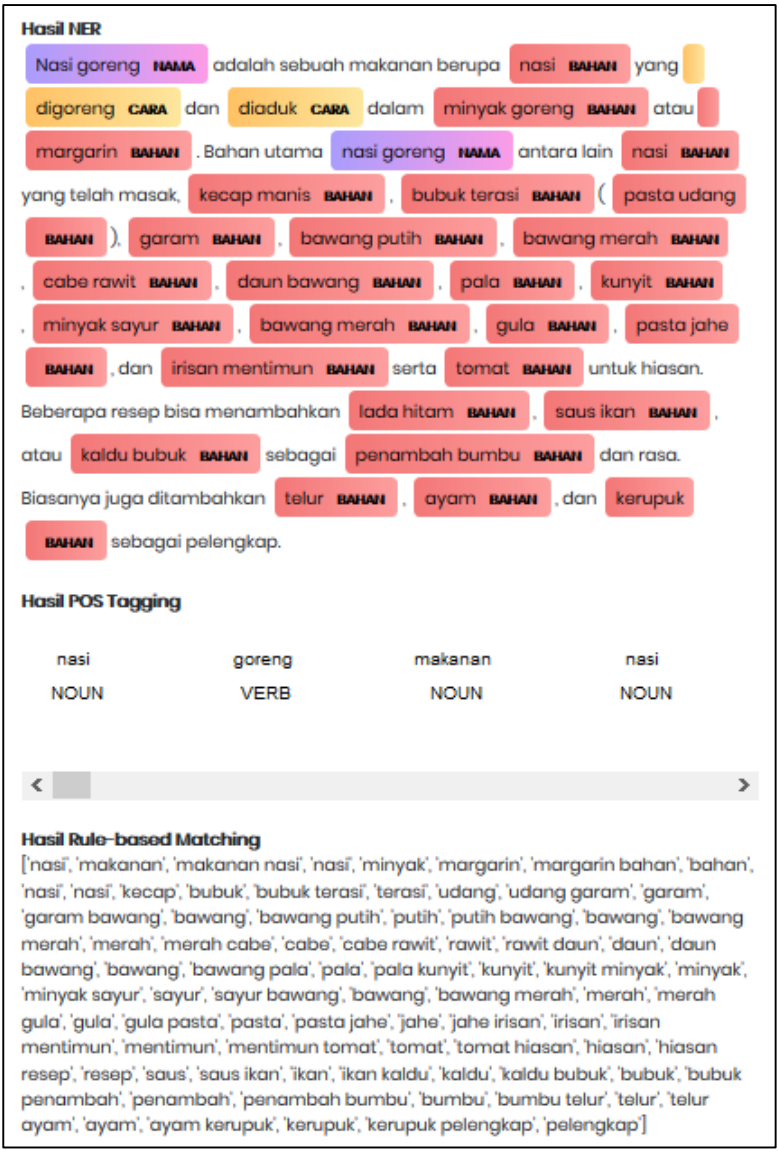
Gambar 5.3 Hasil *input* Data Uji 4 yang tersimpan pada Apache Jena Fuseki



Gambar 5.4 Hasil uji coba *input* data menggunakan Data Uji 5

	subject	property	object
1	ont:krengsengan	rdf:type	ont:makanan
2	ont:krengsengan	ont:bahan	"tongseng"
3	ont:krengsengan	ont:bahan	"bawang putih"
4	ont:krengsengan	ont:bahan	"daging kambing"
5	ont:krengsengan	ont:bahan	"bawang merah"
6	ont:krengsengan	ont:bahan	"ketumbar"
7	ont:krengsengan	ont:bahan	"merica"
8	ont:krengsengan	ont:bahan	"pala"
9	ont:krengsengan	ont:bahan	"kecap manis"
10	ont:krengsengan	ont:bahan	"jeroan kambing"
11	ont:krengsengan	ont:bahan	"gulai kambing"
12	ont:krengsengan	ont:cara	"dihaluskan"
13	ont:krengsengan	ont:deskripsi	"Krengsengan adalah masakan yang biasanya terbuat dari daging kambing yang dipotong kecil (2X2 cm), yang diberi bumbu tumisan bawang merah, bawang putih, ketumbar, merica, pala, (dihaluskan terlebih dahulu) ditambah dengan kecap manis pada waktu memasak, sehingga tidak banyak berkuah dan berwarna coklat tua, ada juga yang dicampur dengan bagian jeroan kambing. Masakan yang khas menggunakan daging kambing ini biasanya dijumpai bersama gulai kambing, dan tongseng yang juga menggunakan daging kambing. Masakan ini umumnya dikenal di Jawa Timur, Jawa Tengah dan DI Yogyakarta."
14	ont:krengsengan	rdfs:label	"krengsengan"

Gambar 5.5 Hasil *input* Data Uji 5 yang tersimpan pada Apache Jena Fuseki



Gambar 5.6 Hasil uji coba *input* data menggunakan Data Uji 6

	subject	property	object
1	ont:nasi_goreng	rdf:type	ont:makanan
2	ont:nasi_goreng	ont:bahan	"bawang putih"
3	ont:nasi_goreng	ont:bahan	"kunyit"
4	ont:nasi_goreng	ont:bahan	"garam"
5	ont:nasi_goreng	ont:bahan	"telur"
6	ont:nasi_goreng	ont:bahan	"tomat"
7	ont:nasi_goreng	ont:bahan	"ayam"
8	ont:nasi_goreng	ont:bahan	"gula"
9	ont:nasi_goreng	ont:bahan	"bawang merah"
10	ont:nasi_goreng	ont:bahan	"daun bawang"
11	ont:nasi_goreng	ont:bahan	"kerupuk"
12	ont:nasi_goreng	ont:bahan	"pala"
13	ont:nasi_goreng	ont:bahan	"nasi"
14	ont:nasi_goreng	ont:bahan	"minyak goreng"
15	ont:nasi_goreng	ont:bahan	"cabe rawit"
16	ont:nasi_goreng	ont:bahan	"margarin"
17	ont:nasi_goreng	ont:bahan	"kecap manis"
18	ont:nasi_goreng	ont:bahan	"irisan mentimun"
19	ont:nasi_goreng	ont:bahan	"bubuk terasi"

Gambar 5.7 Hasil *input* Data Uji 6 yang tersimpan pada Apache Jena Fuseki (1)

20	ont:nasi_goreng	ont:bahan	"pasta udang"
21	ont:nasi_goreng	ont:bahan	"minyak sayur"
22	ont:nasi_goreng	ont:bahan	"pasta jahe"
23	ont:nasi_goreng	ont:bahan	"lada hitam"
24	ont:nasi_goreng	ont:bahan	"saus ikan"
25	ont:nasi_goreng	ont:bahan	"kaldu bubuk"
26	ont:nasi_goreng	ont:bahan	"penambah bumbu"
27	ont:nasi_goreng	ont:cara	"digoreng"
28	ont:nasi_goreng	ont:cara	"diaduk"
29	ont:nasi_goreng	ont:deskripsi	"Nasi goreng adalah sebuah makanan berupa nasi yang digoreng dan diaduk dalam minyak goreng atau margarin. Bahan utama nasi goreng antara lain nasi yang telah masak, kecap manis, bubuk terasi (pasta udang), garam, bawang putih, bawang merah, cabe rawit, daun bawang, pala, kunyit, minyak sayur, bawang merah, gula, pasta jahe, dan irisan mentimun serta tomat untuk hiasan. Beberapa resep bisa menambahkan lada hitam, saus ikan, atau kaldu bubuk sebagai penambah bumbu dan rasa. Biasanya juga ditambahkan telur, ayam, dan kerupuk sebagai pelengkap."
30	ont:nasi_goreng	rdfs:label	"nasi goreng"

Gambar 5.8 Hasil *input* Data Uji 6 yang tersimpan pada Apache Jena Fuseki (2)

Pada saat uji coba *input* data menggunakan Data Uji 4 dan Data Uji 5 tidak ada entitas nama yang terdeteksi. Selain itu, pada saat uji coba *input* data menggunakan Data Uji 5 terdapat kesalahan pendeteksian label, yaitu entitas yang seharusnya terdeteksi sebagai nama makanan justru terdeteksi sebagai bahan. Sedangkan pada saat uji coba *input* data menggunakan Data Uji 6, semua entitas sudah terdeteksi dengan benar. Keseluruhan hasil deteksi NER tersebut berhasil di-*insert* ke *triple store* Apache Jena Fuseki.

5.3.3 Skenario Pengujian 3

Pada skenario ini dilakukan uji coba *query* label dengan memasukkan kata kunci yang berupa nama, bahan, rasa, atau cara

memasak makanan. Uji coba *query* label dilakukan terhadap 3.910 pasang *triplets* yang tersimpan dalam *triple store* Apache Jena Fuseki seperti yang telah dijelaskan pada subbab 4.3.7. Contoh data kata kunci yang akan diuji terdapat pada Tabel 5.6. Hasil uji coba *query* label dapat dilihat pada Tabel 5.7. Uji coba *query* label berhasil menampilkan data makanan yang sesuai dengan kata kunci pencarian baik berupa nama, bahan, rasa, cara memasak, maupun gabungan dari berbagai entitas tersebut.

Tabel 5.6 Data uji coba untuk *query* label

Nama	Data	Keterangan
Data Uji 7	bakpia	Kata kunci pencarian berupa nama makanan
Data Uji 8	tempe	Kata kunci pencarian berupa bahan
Data Uji 9	pahit	Kata kunci pencarian berupa rasa
Data Uji 10	panggang	Kata kunci pencarian berupa cara memasak
Data Uji 11	gula gurih	Kata kunci pencarian berupa bahan dan rasa
Data Uji 12	ikan pedas bakar	Kata kunci pencarian berupa bahan, rasa, dan cara memasak

Tabel 5.7 Hasil uji coba *query* label

Nama	Kata Kunci Pencarian	Hasil Pencarian
Data Uji 7	bakpia	bakpia pathuk bakpia
Data Uji 8	tempe	ketoprak soto tempe penyet

Nama	Kata Kunci Pencarian	Hasil Pencarian
		nasi langi mendoan nasi uduk gudeg nasi kucing keripik tempe sayur lodeh semur lentog garang asem nasi jinggo krecek nasi liwet rujak cingur gado-gado betawi nasi boranan nasi jenggo brongkos
Data Uji 9	pahit	ulukutek leunca semur jengkol
Data Uji 10	panggang	bingka bolu gulung pukis sate madura pisang gapit lapis legit bagiak bakpia sate klathak kue rangi ayam betutu keripik kentang otak-otak kue bandros

Nama	Kata Kunci Pencarian	Hasil Pencarian
		sate lilit kue sus bakpia pathuk bikang kemplang
Data Uji 11	gula gurih	kicak karedok dawet kue rangi srabi solo cendol kerak telor urap enting-enting kude adrem bikang gatot gulai dodongkal surabi
Data Uji 12	ikan pedas bakar	ikan rica

5.3.4 Skenario Pengujian 4

Pada skenario ini dilakukan uji coba bentuk anotasi dari Apache Jena Fuseki yang dapat diunduh dalam *file* berformat JSON. Hasil uji coba bentuk anotasi menggunakan Data Uji 4 dapat dilihat pada Lampiran 4. Hasil uji coba bentuk anotasi menggunakan data Data Uji 5 dapat dilihat pada Lampiran 5. Hasil uji coba bentuk anotasi menggunakan Data Uji 6 dapat dilihat pada Lampiran 6.

5.3.5 Skenario Pengujian 5

Pada skenario ini dilakukan uji coba aplikasi oleh *end user*. *End user* yang dimaksud adalah semua pengguna aplikasi yang ingin mencari informasi tentang makanan atau kuliner sesuai dengan keinginannya. Daftar *end user* yang terlibat dalam pengujian aplikasi dapat dilihat pada Tabel 5.8.

Setelah dilakukan pengujian, *end user* akan diminta mengisi kuesioner untuk menilai aplikasi berdasarkan kebermanfaatannya. Kuesioner terdiri dari dua pertanyaan dengan jawaban berupa nilai dari rentang 1 sampai 5. Interpretasi nilai tersebut secara kualitatif ditunjukkan pada Tabel 5.9. Hasil kuesioner dapat dilihat pada Tabel 5.10. Dari kuesioner tersebut, didapatkan bahwa responden mengatakan bahwa aplikasi ini bermanfaat dengan nilai persentase sebesar 90%.

Tabel 5.8 Daftar *end user* pada pengujian aplikasi

Nama	Keterangan
Pengguna 1	Ibu rumah tangga
Pengguna 2	Penyedia jasa layanan *catering*
Pengguna 3	Mahasiswa
Pengguna 4	Pekerja kantoran
Pengguna 5	Mahasiswa

Tabel 5.9 Interpretasi nilai secara kualitatif

Nilai	Keterangan
1	Sangat kurang
2	Kurang
3	Cukup
4	Baik
5	Sangat baik

Tabel 5.10 Hasil kuesioner

No.	Pertanyaan	Nilai					Total	Persentase
		1	2	3	4	5		
1	Apakah hasil pencarian yang ditampilkan sistem sesuai dengan keinginan Anda?				2	3	23	92%
2	Apakah sistem bermanfaat dan dapat membantu Anda dalam mencari kuliner yang Anda inginkan?			1	1	3	22	88%
Rata-rata								90%

5.4. Evaluasi

Pada Skenario Pengujian 1 diperoleh hasil terbaik saat menggunakan Data Uji 1 dengan rata-rata nilai *accuracy* 94.78%, *precision* 89.57%, *recall* 90.36%, dan *f-measure* 89.80% untuk semua entitas NER.

Pada Skenario Pengujian 2 diperoleh bahwa sistem berhasil mendeteksi entitas NER, *part-of-speech* setiap kata, dan entitas sesuai dengan *rule* yang telah dibuat. Sistem juga berhasil menyimpan hasil anotasi NER dalam bentuk pasangan *triplets* pada Apache Jena Fuseki. Sistem otomatis menambahkan data masukan baru ke *triple store.* Namun, apabila ternyata data masukan baru tersebut nama makanannya sudah ada pada *triple store* tetapi dengan deskripsi makanan yang berbeda, maka makanan tersebut tetap akan memiliki satu nama makanan (tidak

redundan) dengan lebih dari satu deskripsi yang berbeda. Jadi pada intinya sistem pasti akan menyimpan masukan baru yang sebelumnya memang belum ada dalam *triple store.*

Pada Skenario Pengujian 3 diperoleh bahwa sistem berhasil menampilkan data makanan yang sesuai dengan kata kunci pencarian yang dimasukkan pengguna ke dalam sistem.

Pada Skenario Pengujian 4 diperoleh bahwa sistem berhasil menyimpan bentuk anotasi dari Apache Jena Fuseki ke dalam format JSON dan pengguna dapat mengunduh *file* JSON tersebut.

Pada Skenario Pengujian 5 diperoleh bahwa menurut *end user*, aplikasi pencarian kuliner yang dibuat pada tugas akhir ini bermanfaat dengan nilai persentase sebesar 90%.

BAB VI
KESIMPULAN DAN SARAN

Pada bab ini akan diberikan kesimpulan yang diperoleh selama pengerjaan tugas akhir dan saran mengenai pengembangan yang dapat dilakukan terhadap tugas akhir ini di masa yang akan datang.

6.1 Kesimpulan

Dari hasil pengamatan selama proses perancangan, implementasi, dan pengujian yang dilakukan, dapat diambil kesimpulan sebagai berikut:

1. Ekstraksi setiap anotasi pada deskripsi makanan menjadi sebuah metadata dilakukan dengan mendeteksi *named entity* berupa nama, bahan, rasa, dan cara memasak makanan menggunakan model NER. Hasil anotasi NER disimpan dalam bentuk pasangan *triplets* pada *triple store* Apache Jena Fuseki. Hasil anotasi tersebut juga dapat diunduh dari Apache Jena Fuseki dalam format JSON untuk memastikan bahwa bentuk anotasi sudah sesuai dengan standar.
2. Metadata makanan dimodelkan dalam bentuk pasangan *triplets* pada *triple store* Apache Jena Fuseki. Metadata tersebut dapat digunakan untuk menjawab *query* SPARQL tentang nama, bahan, rasa, dan cara memasak makanan.

6.2 Saran

Berikut merupakan beberapa saran untuk pengembangan sistem di masa yang akan datang. Saran-saran ini didasarkan pada hasil perancangan, implementasi, dan pengujian yang telah dilakukan.

1. Menambah jumlah data latih yang digunakan untuk pembuatan model NER agar model NER yang dibuat mampu mendeteksi lebih banyak entitas yang beragam.
2. Data latih yang digunakan untuk pembuatan model POS *Tagging* disesuaikan dengan topik permasalahan agar model

POS *Tagging* dapat memberikan hasil yang lebih akurat dalam mendeteksi kelas kata.

3. Memperluas batasan dan ruang lingkup pembuatan model NER dengan menambahkan entitas baru, seperti asal dan kategori makanan agar korpus data makanan yang dibuat menjadi lebih lengkap.

DAFTAR PUSTAKA

[1] D. Purnamasari, "Klik, Bagikan, Pesan: Kebiasaan Bersantap di Era Media Sosial - Tirto.ID," 2018. [Online]. Available: https://tirto.id/klik-bagikan-pesan-kebiasaan-bersantap-di-era-media-sosial-cJpJ. [Accessed: 27-May-2019].

[2] W. I. Satyagraha, "Perancangan Ontologi untuk Pemodelan Data pada Virtual Chatbot Assistant Untuk Studi Kasus Aplikasi Gifood.id," Institut Teknologi Sepuluh Nopember, 2019.

[3] S. Nugroho, "Virtual Assistant Chatbot pada Aplikasi Speech Recognition dengan Natural Language Processing," Institut Teknologi Sepuluh Nopember, 2019.

[4] "What is Corpus?" [Online]. Available: http://language.worldofcomputing.net/linguistics/introduction/what-is-corpus.html. [Accessed: 01-Jan-2020].

[5] A. Mansouri, L. S. Affendey, and A. Mamat, "Named Entity Recognition Approaches," *J. Comput. Sci.*, vol. 8, no. 2, pp. 339–344, 2008.

[6] A. Mulyanto, Y. A. Nurhuda, and N. Wiyanto, "Penyelesaian Kata Ambigu Pada Proses POS Tagging Menggunakan Algoritma Hidden markov Model (HMM)," no. 978, pp. 347–358, 2017.

[7] A. KS, "Rule-Based Matching with spaCy - Ashiq KS - Medium," 2019. [Online]. Available: https://medium.com/@ashiqgiga07/rule-based-matching-with-spacy-295b76ca2b68. [Accessed: 17-Dec-2019].

[8] "spaCy 101: Everything you need to know · spaCy Usage Documentation." [Online]. Available: https://spacy.io/usage/spacy-101.[Accessed:17-Dec-2019].

[9] “Prodigy · An annotation tool for AI, Machine Learning & NLP.” [Online]. Available: https://prodi.gy/. [Accessed: 17-Dec-2019].

[10] “Flask · PyPI.” [Online]. Available: https://pypi.org/project/Flask/. [Accessed: 17-Dec-2019].

[11] “Apache Jena - Apache Jena Fuseki.” [Online]. Available: https://jena.apache.org/documentation/fuseki2/. [Accessed: 17-Dec-2019].

[12] “Understanding Confusion Matrix - Towards Data Science.” [Online]. Available: https://towardsdatascience.com/understanding-confusion-matrix-a9ad42dcfd62. [Accessed: 26-Dec-2019].

[13] “Web Scraping Tool & Free Web Crawlers | Octoparse.” [Online]. Available: https://www.octoparse.com/. [Accessed: 17-Dec-2019].

[14] “GitHub - famrashel/idn-tagged-corpus: Indonesian Manually Tagged Corpus.” [Online]. Available: https://github.com/famrashel/idn-tagged-corpus. [Accessed: 13-Jan-2020].

[15] “Universal POS tags.” [Online]. Available: https://universaldependencies.org/docs/u/pos/index.html. [Accessed: 17-Dec-2019].

[16] “GitHub - ManivannanMurugavel/spacy-ner-annotator: Train Spacy ner with custom dataset.” [Online]. Available: https://github.com/ManivannanMurugavel/spacy-ner-annotator. [Accessed: 13-Jan-2020].

[17] “protégé.” [Online]. Available: https://protege.stanford.edu/. [Accessed: 26-Dec-2019].

LAMPIRAN

1. Contoh Data Makanan

Berikut contoh data makanan yang digunakan pada tugas akhir ini.

Nama Makanan	Deskripsi Makanan
Bandeng presto	Bandeng presto adalah makanan khas Indonesia yang berasal dari Kota Semarang, Jawa Tengah. Makanan ini dibuat dari ikan bandeng yang dibumbui dengan bawang putih, kunyit dan garam. Ikan bandeng ini kemudian dimasak pada alas daun pisang dengan cara presto.
Tahu campur	Tahu campur adalah salah satu makanan khas Jawa Timur. Tahu campur terdiri dari sop daging sapi kenyal, tahu goreng, perkedel singkong, taoge segar, selada air segar, mi kuning, dan kerupuk udang. Semua ini kemudian dicampurkan ke bumbu petis, bawang goreng, dan sambal.
Kue putu	Kue putu adalah jenis makanan tradisional nusantara yang berupa kue dengan isian gula jawa, dibalut dengan parutan kelapa, dan tepung beras butiran kasar. Kue ini dikukus dengan diletakkan di dalam tabung bambu yang sedikit dipadatkan.

2. *File* RDF/XML dari Protégé

Berikut isi file RDF/XML dari Protégé yang diunggah pada Apache Jena Fuseki sebagai data awal.

```
1.  <?xml version="1.0"?>
2.  <rdf:RDF xmlns="http://food-ontology/3#"
3.       xml:base="http://food-ontology/3"
```

```
4.      xmlns:food-ontology="http://food-
    ontology/3#"
5.      xmlns:rdf="http://www.w3.org/1999/02/22-rdf-
    syntax-ns#"
6.      xmlns:owl="http://www.w3.org/2002/07/owl#"
7.      xmlns:xml="http://www.w3.org/XML/1998/namesp
    ace"
8.      xmlns:xsd="http://www.w3.org/2001/XMLSchema#
    "
9.      xmlns:rdfs="http://www.w3.org/2000/01/rdf-
    schema#">
10.    <owl:Ontology rdf:about="http://food-
    ontology/3"/>
11.
12.    <!--
13.    ///////////////////////////////////////////////
    ////////////////////////////////////////////
14.    //
15.    // Annotation properties
16.    //
17.    ///////////////////////////////////////////////
    ////////////////////////////////////////////
18.     -->
19.
20.    <!-- http://food-ontology/3#bahan -->
21.
22.    <owl:AnnotationProperty rdf:about="http://foo
    d-ontology/3#bahan"/>
23.
24.
25.    <!-- http://food-ontology/3#cara -->
26.
27.    <owl:AnnotationProperty rdf:about="http://foo
    d-ontology/3#cara"/>
28.
29.
30.    <!-- http://food-ontology/3#deskripsi -->
31.
32.    <owl:AnnotationProperty rdf:about="http://foo
    d-ontology/3#deskripsi"/>
33.
34.
```

```
35.     <!-- http://food-ontology/3#rasa -->
36.
37.     <owl:AnnotationProperty rdf:about="http://food-ontology/3#rasa"/>
38.
39.
40.     <!--
41.     ///////////////////////////////////////////////////////////////////////////////////////////
42.     //
43.     // Classes
44.     //
45.     ///////////////////////////////////////////////////////////////////////////////////////////
46.      -->
47.
48.     <!-- http://food-ontology/3#makanan -->
49.
50.     <owl:Class rdf:about="http://food-ontology/3#makanan"/>
51. </rdf:RDF>
52.
53.
54. <!-- Generated by the OWL API (version 4.2.8.20170104-2310) https://github.com/owlcs/owlapi -->
```

3. *File* JSON *Output* Hasil Anotasi dari Apache Jena Fuseki

Berikut contoh *file* JSON yang berisi hasil anotasi yang diunduh dari Apache Jena Fuseki.


```
{"head": {"vars": ["subject", "property", "object", "property2", "type"]}, "results": {"bindings": [{"subject": {"type": "uri", "value": "http://food-ontology/3#kue_putu"}, "property": {"type": "uri", "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#type"}, "object": {"type": "uri", "value": "http://food-ontology/3#makanan"}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#kue_putu"},
```

"property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "gula jawa"}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#kue_putu"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "tepung beras"}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#kue_putu"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "parutan kelapa"}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#kue_putu"}, "property": {"type": "uri", "value": "http://food-ontology/3#cara"}, "object": {"type": "literal", "value": "dikukus"}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#kue_putu"}, "property": {"type": "uri", "value": "http://food-ontology/3#deskripsi"}, "object": {"type": "literal", "value": "Kue putu adalah jenis makanan tradisional nusantara yang berupa kue dengan isian gula jawa, dibalut dengan parutan kelapa, dan tepung beras butiran kasar. Kue ini dikukus dengan diletakkan di dalam tabung bambu yang sedikit dipadatkan."}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#kue_putu"}, "property": {"type": "uri", "value": "http://www.w3.org/2000/01/rdf-schema#label"}, "object": {"type": "literal", "value": "kue putu"}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#kue_putu"}, "property": {"type": "uri", "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#type"}, "object": {"type": "uri", "value": "http://food-ontology/3#makanan"}, "property2": {"type": "uri", "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#type"}, "type": {"type": "uri", "value": "http://www.w3.org/2002/07/owl#Class"}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#kue_putu"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "gula jawa"}, "property2": {"type": "uri", "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#type"}, "type": {"type": "uri", "value": "http://www.w3.org/2002/07/owl#AnnotationProperty"}}, {"subject": {"type": "uri", "value": "http://food-

```
ontology/3#kue_putu"},   "property":   {"type":   "uri",
"value":    "http://food-ontology/3#bahan"},    "object":
{"type":    "literal",    "value":    "tepung    beras"},
"property2":        {"type":        "uri",        "value":
"http://www.w3.org/1999/02/22-rdf-syntax-ns#type"},
"type":          {"type":          "uri",          "value":
"http://www.w3.org/2002/07/owl#AnnotationProperty"}},
{"subject":  {"type":  "uri",  "value":  "http://food-
ontology/3#kue_putu"},   "property":   {"type":   "uri",
"value":    "http://food-ontology/3#bahan"},    "object":
{"type":   "literal",   "value":   "parutan   kelapa"},
"property2":        {"type":        "uri",        "value":
"http://www.w3.org/1999/02/22-rdf-syntax-ns#type"},
"type":          {"type":          "uri",          "value":
"http://www.w3.org/2002/07/owl#AnnotationProperty"}},
{"subject":  {"type":  "uri",  "value":  "http://food-
ontology/3#kue_putu"},   "property":   {"type":   "uri",
"value":    "http://food-ontology/3#cara"},    "object":
{"type":  "literal",  "value":  "dikukus"},  "property2":
{"type":  "uri",  "value":  "http://www.w3.org/1999/02/22-
rdf-syntax-ns#type"},  "type":  {"type":  "uri",  "value":
"http://www.w3.org/2002/07/owl#AnnotationProperty"}},
{"subject":  {"type":  "uri",  "value":  "http://food-
ontology/3#kue_putu"},   "property":   {"type":   "uri",
"value":  "http://food-ontology/3#deskripsi"},  "object":
{"type":  "literal",  "value":  "Kue  putu  adalah  jenis
makanan tradisional nusantara yang berupa kue dengan isian
gula jawa, dibalut dengan parutan kelapa, dan tepung beras
butiran kasar. Kue ini dikukus dengan diletakkan di dalam
tabung  bambu  yang  sedikit  dipadatkan."},  "property2":
{"type":  "uri",  "value":  "http://www.w3.org/1999/02/22-
rdf-syntax-ns#type"},  "type":  {"type":  "uri",  "value":
"http://www.w3.org/2002/07/owl#AnnotationProperty"}}]}}
```


4. Hasil uji coba bentuk anotasi menggunakan Data Uji 4

Berikut *file* JSON yang berisi hasil anotasi yang diunduh dari Apache Jena Fuseki sebagai hasil uji coba bentuk anotasi menggunakan Data Uji 4.

{"head": {"vars": ["subject", "property", "object", "property2", "type"]}, "results": {"bindings": [{"subject": {"type": "uri", "value": "http://food-ontology/3#kue_dadar_gulung"}, "property": {"type": "uri", "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#type"}, "object": {"type": "uri", "value": "http://food-ontology/3#makanan"}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#kue_dadar_gulung"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "parutan kelapa"}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#kue_dadar_gulung"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "gula jawa cair"}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#kue_dadar_gulung"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "pewarna daun suji"}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#kue_dadar_gulung"}, "property": {"type": "uri", "value": "http://food-ontology/3#deskripsi"}, "object": {"type": "literal", "value": "Kue dadar gulung merupakan penganan khas Indonesia yang dapat digolongkan sebagai panekuk yang diisi dengan parutan kelapa yang dicampur dengan gula jawa cair. Isi ini disebut unti. Kulit dadar gulung berwarna hijau karena diberi pewarna daun suji."}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#kue_dadar_gulung"}, "property": {"type": "uri", "value": "http://www.w3.org/2000/01/rdf-schema#label"}, "object": {"type": "literal", "value": "kue dadar gulung"}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#kue_dadar_gulung"}, "property": {"type": "uri", "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#type"}, "object": {"type": "uri", "value": "http://food-ontology/3#makanan"}, "property2": {"type": "uri", "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#type"}, "type": {"type": "uri", "value": "http://www.w3.org/2002/07/owl#Class"}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#kue_dadar_gulung"}, "property": {"type":

```
"uri",      "value":      "http://food-ontology/3#bahan"},
"object": {"type": "literal", "value": "parutan kelapa"},
"property2":        {"type":        "uri",        "value":
"http://www.w3.org/1999/02/22-rdf-syntax-ns#type"},
"type":         {"type":         "uri",         "value":
"http://www.w3.org/2002/07/owl#AnnotationProperty"}},
{"subject":  {"type":  "uri",  "value":  "http://food-
ontology/3#kue_dadar_gulung"},    "property":    {"type":
"uri",      "value":      "http://food-ontology/3#bahan"},
"object": {"type": "literal", "value": "gula jawa cair"},
"property2":        {"type":        "uri",        "value":
"http://www.w3.org/1999/02/22-rdf-syntax-ns#type"},
"type":         {"type":         "uri",         "value":
"http://www.w3.org/2002/07/owl#AnnotationProperty"}},
{"subject":  {"type":  "uri",  "value":  "http://food-
ontology/3#kue_dadar_gulung"},    "property":    {"type":
"uri",      "value":      "http://food-ontology/3#bahan"},
"object":  {"type":  "literal",  "value":  "pewarna  daun
suji"},    "property2":    {"type":    "uri",    "value":
"http://www.w3.org/1999/02/22-rdf-syntax-ns#type"},
"type":         {"type":         "uri",         "value":
"http://www.w3.org/2002/07/owl#AnnotationProperty"}},
{"subject":  {"type":  "uri",  "value":  "http://food-
ontology/3#kue_dadar_gulung"},    "property":    {"type":
"uri",   "value":   "http://food-ontology/3#deskripsi"},
"object": {"type": "literal", "value": "Kue dadar gulung
merupakan penganan khas Indonesia yang dapat digolongkan
sebagai panekuk yang diisi dengan parutan kelapa yang
dicampur dengan gula jawa cair. Isi ini disebut unti.
Kulit dadar gulung berwarna hijau karena diberi pewarna
daun  suji."},  "property2":  {"type":  "uri",  "value":
"http://www.w3.org/1999/02/22-rdf-syntax-ns#type"},
"type":         {"type":         "uri",         "value":
"http://www.w3.org/2002/07/owl#AnnotationProperty"}}]}}
```


5. Hasil uji coba bentuk anotasi menggunakan Data Uji 5

Berikut *file* JSON yang berisi hasil anotasi yang diunduh dari Apache Jena Fuseki sebagai hasil uji coba bentuk anotasi menggunakan Data Uji 5.

{"head": {"vars": ["subject", "property", "object", "property2", "type"]}, "results": {"bindings": [{"subject": {"type": "uri", "value": "http://food-ontology/3#krengsengan"}, "property": {"type": "uri", "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#type"}, "object": {"type": "uri", "value": "http://food-ontology/3#makanan"}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#krengsengan"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "tongseng"}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#krengsengan"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "bawang putih"}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#krengsengan"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "daging kambing"}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#krengsengan"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "bawang merah"}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#krengsengan"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "ketumbar"}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#krengsengan"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "merica"}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#krengsengan"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "pala"}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#krengsengan"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "kecap manis"}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#krengsengan"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "jeroan kambing"}},

{"subject": {"type": "uri", "value": "http://food-ontology/3#krengsengan"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "gulai kambing"}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#krengsengan"}, "property": {"type": "uri", "value": "http://food-ontology/3#cara"}, "object": {"type": "literal", "value": "dihaluskan"}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#krengsengan"}, "property": {"type": "uri", "value": "http://food-ontology/3#deskripsi"}, "object": {"type": "literal", "value": "Krengsengan adalah masakan yang biasanya terbuat dari daging kambing yang dipotong kecil (2X2 cm), yang diberi bumbu tumisan bawang merah, bawang putih, ketumbar, merica, pala, (dihaluskan terlebih dahulu) ditambah dengan kecap manis pada waktu memasak, sehingga tidak banyak berkuah dan berwarna coklat tua, ada juga yang dicampur dengan bagian jeroan kambing. Masakan yang khas menggunakan daging kambing ini biasanya dijumpai bersama gulai kambing, dan tongseng yang juga menggunakan daging kambing. Masakan ini umumnya dikenal di Jawa Timur, Jawa Tengah dan DI Yogyakarta."}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#krengsengan"}, "property": {"type": "uri", "value": "http://www.w3.org/2000/01/rdf-schema#label"}, "object": {"type": "literal", "value": "krengsengan"}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#krengsengan"}, "property": {"type": "uri", "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#type"}, "object": {"type": "uri", "value": "http://food-ontology/3#makanan"}, "property2": {"type": "uri", "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#type"}, "type": {"type": "uri", "value": "http://www.w3.org/2002/07/owl#Class"}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#krengsengan"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "tongseng"}, "property2": {"type": "uri", "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#type"}, "type": {"type": "uri", "value": "http://www.w3.org/2002/07/owl#AnnotationProperty"}}, {"subject": {"type": "uri", "value": "http://food-

ontology/3#krengsengan"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "bawang putih"}, "property2": {"type": "uri", "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#type"}, "type": {"type": "uri", "value": "http://www.w3.org/2002/07/owl#AnnotationProperty"}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#krengsengan"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "daging kambing"}, "property2": {"type": "uri", "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#type"}, "type": {"type": "uri", "value": "http://www.w3.org/2002/07/owl#AnnotationProperty"}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#krengsengan"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "bawang merah"}, "property2": {"type": "uri", "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#type"}, "type": {"type": "uri", "value": "http://www.w3.org/2002/07/owl#AnnotationProperty"}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#krengsengan"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "ketumbar"}, "property2": {"type": "uri", "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#type"}, "type": {"type": "uri", "value": "http://www.w3.org/2002/07/owl#AnnotationProperty"}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#krengsengan"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "merica"}, "property2": {"type": "uri", "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#type"}, "type": {"type": "uri", "value": "http://www.w3.org/2002/07/owl#AnnotationProperty"}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#krengsengan"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "pala"}, "property2": {"type": "uri", "value": "http://www.w3.org/1999/02/22-

rdf-syntax-ns#type"}, "type": {"type": "uri", "value": "http://www.w3.org/2002/07/owl#AnnotationProperty"}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#krengsengan"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "kecap manis"}, "property2": {"type": "uri", "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#type"}, "type": {"type": "uri", "value": "http://www.w3.org/2002/07/owl#AnnotationProperty"}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#krengsengan"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "jeroan kambing"}, "property2": {"type": "uri", "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#type"}, "type": {"type": "uri", "value": "http://www.w3.org/2002/07/owl#AnnotationProperty"}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#krengsengan"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "gulai kambing"}, "property2": {"type": "uri", "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#type"}, "type": {"type": "uri", "value": "http://www.w3.org/2002/07/owl#AnnotationProperty"}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#krengsengan"}, "property": {"type": "uri", "value": "http://food-ontology/3#cara"}, "object": {"type": "literal", "value": "dihaluskan"}, "property2": {"type": "uri", "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#type"}, "type": {"type": "uri", "value": "http://www.w3.org/2002/07/owl#AnnotationProperty"}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#krengsengan"}, "property": {"type": "uri", "value": "http://food-ontology/3#deskripsi"}, "object": {"type": "literal", "value": "Krengsengan adalah masakan yang biasanya terbuat dari daging kambing yang dipotong kecil (2X2 cm), yang diberi bumbu tumisan bawang merah, bawang putih, ketumbar, merica, pala, (dihaluskan terlebih dahulu) ditambah dengan kecap manis pada waktu memasak, sehingga tidak banyak berkuah dan berwarna coklat tua, ada juga yang dicampur dengan bagian jeroan kambing. Masakan

```
yang khas menggunakan daging kambing ini biasanya dijumpai bersama gulai kambing, dan tongseng yang juga menggunakan daging kambing. Masakan ini umumnya dikenal di Jawa Timur, Jawa Tengah dan DI Yogyakarta."}, "property2": {"type": "uri", "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#type"}, "type": {"type": "uri", "value": "http://www.w3.org/2002/07/owl#AnnotationProperty"}}]}}
```


6. Hasil uji coba bentuk anotasi menggunakan Data Uji 6

Berikut *file* JSON yang berisi hasil anotasi yang diunduh dari Apache Jena Fuseki sebagai hasil uji coba bentuk anotasi menggunakan Data Uji 6.


```
{"head": {"vars": ["subject", "property", "object", "property2", "type"]}, "results": {"bindings": [{"subject": {"type": "uri", "value": "http://food-ontology/3#nasi_goreng"}, "property": {"type": "uri", "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#type"}, "object": {"type": "uri", "value": "http://food-ontology/3#makanan"}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#nasi_goreng"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "bawang putih"}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#nasi_goreng"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "kunyit"}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#nasi_goreng"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "garam"}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#nasi_goreng"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "telur"}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#nasi_goreng"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "tomat"}}, {"subject": {"type": "uri", "value": "http://food-
```

```
ontology/3#nasi_goreng"},  "property":  {"type":  "uri",
"value":    "http://food-ontology/3#bahan"},    "object":
{"type":   "literal",   "value":   "ayam"}},   {"subject":
{"type":        "uri",        "value":        "http://food-
ontology/3#nasi_goreng"},  "property":  {"type":  "uri",
"value":    "http://food-ontology/3#bahan"},    "object":
{"type":   "literal",   "value":   "gula"}},   {"subject":
{"type":        "uri",        "value":        "http://food-
ontology/3#nasi_goreng"},  "property":  {"type":  "uri",
"value":    "http://food-ontology/3#bahan"},    "object":
{"type":    "literal",    "value":    "bawang    merah"}},
{"subject":   {"type":   "uri",   "value":   "http://food-
ontology/3#nasi_goreng"},  "property":  {"type":  "uri",
"value":    "http://food-ontology/3#bahan"},    "object":
{"type": "literal", "value": "daun bawang"}}, {"subject":
{"type":        "uri",        "value":        "http://food-
ontology/3#nasi_goreng"},  "property":  {"type":  "uri",
"value":    "http://food-ontology/3#bahan"},    "object":
{"type":  "literal",  "value":  "kerupuk"}},  {"subject":
{"type":        "uri",        "value":        "http://food-
ontology/3#nasi_goreng"},  "property":  {"type":  "uri",
"value":    "http://food-ontology/3#bahan"},    "object":
{"type":   "literal",   "value":   "pala"}},   {"subject":
{"type":        "uri",        "value":        "http://food-
ontology/3#nasi_goreng"},  "property":  {"type":  "uri",
"value":    "http://food-ontology/3#bahan"},    "object":
{"type":   "literal",   "value":   "nasi"}},   {"subject":
{"type":        "uri",        "value":        "http://food-
ontology/3#nasi_goreng"},  "property":  {"type":  "uri",
"value":    "http://food-ontology/3#bahan"},    "object":
{"type":    "literal",    "value":    "minyak    goreng"}},
{"subject":   {"type":   "uri",   "value":   "http://food-
ontology/3#nasi_goreng"},  "property":  {"type":  "uri",
"value":    "http://food-ontology/3#bahan"},    "object":
{"type": "literal", "value": "cabe rawit"}}, {"subject":
{"type":        "uri",        "value":        "http://food-
ontology/3#nasi_goreng"},  "property":  {"type":  "uri",
"value":    "http://food-ontology/3#bahan"},    "object":
{"type":  "literal",  "value":  "margarin"}},  {"subject":
{"type":        "uri",        "value":        "http://food-
ontology/3#nasi_goreng"},  "property":  {"type":  "uri",
"value":    "http://food-ontology/3#bahan"},    "object":
```

{"type": "literal", "value": "kecap manis"}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#nasi_goreng"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "irisan mentimun"}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#nasi_goreng"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "bubuk terasi"}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#nasi_goreng"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "pasta udang"}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#nasi_goreng"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "minyak sayur"}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#nasi_goreng"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "pasta jahe"}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#nasi_goreng"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "lada hitam"}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#nasi_goreng"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "saus ikan"}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#nasi_goreng"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "kaldu bubuk"}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#nasi_goreng"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "penambah bumbu"}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#nasi_goreng"}, "property": {"type": "uri", "value": "http://food-ontology/3#cara"}, "object": {"type": "literal", "value": "digoreng"}}, {"subject": {"type": "uri", "value": "http://food-

ontology/3#nasi_goreng"}, "property": {"type": "uri", "value": "http://food-ontology/3#cara"}, "object": {"type": "literal", "value": "diaduk"}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#nasi_goreng"}, "property": {"type": "uri", "value": "http://food-ontology/3#deskripsi"}, "object": {"type": "literal", "value": "Nasi goreng adalah sebuah makanan berupa nasi yang digoreng dan diaduk dalam minyak goreng atau margarin. Bahan utama nasi goreng antara lain nasi yang telah masak, kecap manis, bubuk terasi (pasta udang), garam, bawang putih, bawang merah, cabe rawit, daun bawang, pala, kunyit, minyak sayur, bawang merah, gula, pasta jahe, dan irisan mentimun serta tomat untuk hiasan. Beberapa resep bisa menambahkan lada hitam, saus ikan, atau kaldu bubuk sebagai penambah bumbu dan rasa. Biasanya juga ditambahkan telur, ayam, dan kerupuk sebagai pelengkap."}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#nasi_goreng"}, "property": {"type": "uri", "value": "http://www.w3.org/2000/01/rdf-schema#label"}, "object": {"type": "literal", "value": "nasi goreng"}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#nasi_goreng"}, "property": {"type": "uri", "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#type"}, "object": {"type": "uri", "value": "http://food-ontology/3#makanan"}, "property2": {"type": "uri", "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#type"}, "type": {"type": "uri", "value": "http://www.w3.org/2002/07/owl#Class"}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#nasi_goreng"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "bawang putih"}, "property2": {"type": "uri", "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#type"}, "type": {"type": "uri", "value": "http://www.w3.org/2002/07/owl#AnnotationProperty"}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#nasi_goreng"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "kunyit"}, "property2": {"type": "uri", "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#type"}, "type": {"type": "uri", "value":

"http://www.w3.org/2002/07/owl#AnnotationProperty"}},
{"subject": {"type": "uri", "value": "http://food-ontology/3#nasi_goreng"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "garam"}, "property2": {"type": "uri", "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#type"}, "type": {"type": "uri", "value": "http://www.w3.org/2002/07/owl#AnnotationProperty"}},
{"subject": {"type": "uri", "value": "http://food-ontology/3#nasi_goreng"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "telur"}, "property2": {"type": "uri", "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#type"}, "type": {"type": "uri", "value": "http://www.w3.org/2002/07/owl#AnnotationProperty"}},
{"subject": {"type": "uri", "value": "http://food-ontology/3#nasi_goreng"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "tomat"}, "property2": {"type": "uri", "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#type"}, "type": {"type": "uri", "value": "http://www.w3.org/2002/07/owl#AnnotationProperty"}},
{"subject": {"type": "uri", "value": "http://food-ontology/3#nasi_goreng"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "ayam"}, "property2": {"type": "uri", "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#type"}, "type": {"type": "uri", "value": "http://www.w3.org/2002/07/owl#AnnotationProperty"}},
{"subject": {"type": "uri", "value": "http://food-ontology/3#nasi_goreng"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "gula"}, "property2": {"type": "uri", "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#type"}, "type": {"type": "uri", "value": "http://www.w3.org/2002/07/owl#AnnotationProperty"}},
{"subject": {"type": "uri", "value": "http://food-ontology/3#nasi_goreng"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "bawang merah"}, "property2": {"type": "uri", "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#type"},

"type": {"type": "uri", "value": "http://www.w3.org/2002/07/owl#AnnotationProperty"}},
{"subject": {"type": "uri", "value": "http://food-ontology/3#nasi_goreng"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "daun bawang"}, "property2": {"type": "uri", "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#type"}, "type": {"type": "uri", "value": "http://www.w3.org/2002/07/owl#AnnotationProperty"}},
{"subject": {"type": "uri", "value": "http://food-ontology/3#nasi_goreng"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "kerupuk"}, "property2": {"type": "uri", "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#type"}, "type": {"type": "uri", "value": "http://www.w3.org/2002/07/owl#AnnotationProperty"}},
{"subject": {"type": "uri", "value": "http://food-ontology/3#nasi_goreng"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "pala"}, "property2": {"type": "uri", "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#type"}, "type": {"type": "uri", "value": "http://www.w3.org/2002/07/owl#AnnotationProperty"}},
{"subject": {"type": "uri", "value": "http://food-ontology/3#nasi_goreng"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "nasi"}, "property2": {"type": "uri", "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#type"}, "type": {"type": "uri", "value": "http://www.w3.org/2002/07/owl#AnnotationProperty"}},
{"subject": {"type": "uri", "value": "http://food-ontology/3#nasi_goreng"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "minyak goreng"}, "property2": {"type": "uri", "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#type"}, "type": {"type": "uri", "value": "http://www.w3.org/2002/07/owl#AnnotationProperty"}},
{"subject": {"type": "uri", "value": "http://food-ontology/3#nasi_goreng"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "cabe rawit"}, "property2":

{"type": "uri", "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#type"}, "type": {"type": "uri", "value": "http://www.w3.org/2002/07/owl#AnnotationProperty"}},
{"subject": {"type": "uri", "value": "http://food-ontology/3#nasi_goreng"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "margarin"}, "property2": {"type": "uri", "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#type"}, "type": {"type": "uri", "value": "http://www.w3.org/2002/07/owl#AnnotationProperty"}},
{"subject": {"type": "uri", "value": "http://food-ontology/3#nasi_goreng"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "kecap manis"}, "property2": {"type": "uri", "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#type"}, "type": {"type": "uri", "value": "http://www.w3.org/2002/07/owl#AnnotationProperty"}},
{"subject": {"type": "uri", "value": "http://food-ontology/3#nasi_goreng"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "irisan mentimun"}, "property2": {"type": "uri", "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#type"}, "type": {"type": "uri", "value": "http://www.w3.org/2002/07/owl#AnnotationProperty"}},
{"subject": {"type": "uri", "value": "http://food-ontology/3#nasi_goreng"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "bubuk terasi"}, "property2": {"type": "uri", "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#type"}, "type": {"type": "uri", "value": "http://www.w3.org/2002/07/owl#AnnotationProperty"}},
{"subject": {"type": "uri", "value": "http://food-ontology/3#nasi_goreng"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "pasta udang"}, "property2": {"type": "uri", "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#type"}, "type": {"type": "uri", "value": "http://www.w3.org/2002/07/owl#AnnotationProperty"}},
{"subject": {"type": "uri", "value": "http://food-ontology/3#nasi_goreng"}, "property": {"type": "uri",

"value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "minyak sayur"}, "property2": {"type": "uri", "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#type"}, "type": {"type": "uri", "value": "http://www.w3.org/2002/07/owl#AnnotationProperty"}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#nasi_goreng"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "pasta jahe"}, "property2": {"type": "uri", "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#type"}, "type": {"type": "uri", "value": "http://www.w3.org/2002/07/owl#AnnotationProperty"}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#nasi_goreng"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "lada hitam"}, "property2": {"type": "uri", "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#type"}, "type": {"type": "uri", "value": "http://www.w3.org/2002/07/owl#AnnotationProperty"}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#nasi_goreng"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "saus ikan"}, "property2": {"type": "uri", "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#type"}, "type": {"type": "uri", "value": "http://www.w3.org/2002/07/owl#AnnotationProperty"}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#nasi_goreng"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "kaldu bubuk"}, "property2": {"type": "uri", "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#type"}, "type": {"type": "uri", "value": "http://www.w3.org/2002/07/owl#AnnotationProperty"}}, {"subject": {"type": "uri", "value": "http://food-ontology/3#nasi_goreng"}, "property": {"type": "uri", "value": "http://food-ontology/3#bahan"}, "object": {"type": "literal", "value": "penambah bumbu"}, "property2": {"type": "uri", "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#type"}, "type": {"type": "uri", "value": "http://www.w3.org/2002/07/owl#AnnotationProperty"}},

{"subject": {"type": "uri", "value": "http://food-ontology/3#nasi_goreng"}, "property": {"type": "uri", "value": "http://food-ontology/3#cara"}, "object": {"type": "literal", "value": "digoreng"}, "property2": {"type": "uri", "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#type"}, "type": {"type": "uri", "value": "http://www.w3.org/2002/07/owl#AnnotationProperty"}},
{"subject": {"type": "uri", "value": "http://food-ontology/3#nasi_goreng"}, "property": {"type": "uri", "value": "http://food-ontology/3#cara"}, "object": {"type": "literal", "value": "diaduk"}, "property2": {"type": "uri", "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#type"}, "type": {"type": "uri", "value": "http://www.w3.org/2002/07/owl#AnnotationProperty"}},
{"subject": {"type": "uri", "value": "http://food-ontology/3#nasi_goreng"}, "property": {"type": "uri", "value": "http://food-ontology/3#deskripsi"}, "object": {"type": "literal", "value": "Nasi goreng adalah sebuah makanan berupa nasi yang digoreng dan diaduk dalam minyak goreng atau margarin. Bahan utama nasi goreng antara lain nasi yang telah masak, kecap manis, bubuk terasi (pasta udang), garam, bawang putih, bawang merah, cabe rawit, daun bawang, pala, kunyit, minyak sayur, bawang merah, gula, pasta jahe, dan irisan mentimun serta tomat untuk hiasan. Beberapa resep bisa menambahkan lada hitam, saus ikan, atau kaldu bubuk sebagai penambah bumbu dan rasa. Biasanya juga ditambahkan telur, ayam, dan kerupuk sebagai pelengkap."}, "property2": {"type": "uri", "value": "http://www.w3.org/1999/02/22-rdf-syntax-ns#type"}, "type": {"type": "uri", "value": "http://www.w3.org/2002/07/owl#AnnotationProperty"}}]}}

BIODATA PENULIS

Fadilla Sukma Alfiani, lahir di Tulungagung, 9 Februari 1999. Penulis menempuh pendidikan mulai SD Negeri Sepatan Tulungagung (2005-2011), SMP Negeri 1 Kauman Tulungagung (2011-2014), SMA Negeri 1 Boyolangu Tulungagung (2014-2016), dan sekarang sedang menempuh pendidikan S1 Teknik Informatika, Institut Teknologi Sepuluh Nopember Surabaya. Penulis aktif dalam organisasi mahasiswa, diantaranya menjadi staf Departemen Kesejahteraaan Mahasiswa Himpunan Mahasiswa Teknik Computer-Informatika (HMTC) ITS. Selain itu, penulis juga aktif dalam berbagai acara kepanitian, salah satunya Schematics ITS sebagai staf web dan kesekretariatan. Penulis berpengalaman sebagai asisten dosen pada mata kuliah Manajemen Basis Data dan Rekayasa Pengetahuan. Dalam menyelesaikan pendidikan sarjana, penulis mengambil rumpun mata kuliah (RMK) Manajemen Informasi (MI). Untuk komunikasi penulis dapat dihubungi melalui surel fadillasukmaal@gmail.com.